## Built in IOC container

### IserviceCollection

* Register application services to built-in IOC Container.
* It just inherits IList<ServiceDescriptor>
* ServiceCollection class implement IserviceCollection interface
* Adding a service in IserviceCollection type instance => actually creates an instance of ServiceDescriptor and add it to list.
* **Existence ?? – to built the DI container. Aur jab yhe fully built ho jata hai, it gets composed to IserviceProvider instance, jo ki use kiya ja skta hai to resolve service.**

### IServiceProvider

* IServiceProvider has GetService method.
* ServiceProvider class implements GetService method and returns registered services with the container.
* Cannot instantiate ServiceProvider class outside DependencyInjection.dll assembly b/c its controller are marked as internal.
* Aap IserviceProvider instance ko inject kr skte ho inside any class.
* Aur tho aur, IapplicationBuilder (via ApplicationServices prop) and HttpContext (RequestServices prop) classes can provide Service Provider as well.
* Yhe IserviceProvider has GetService(Type type) method used to resolve a service.
  + var service = (IFooService)serviceProvider.GetService(typeof(IFooService));
* There are serveral convenience extension methods available,
  + serviceProvider.GetService<IfooService>();

### ServiceCollection**Service**Extensions class

* Includes extension methods related to **service registration** => jo ki use hote hai to add service with service lifetime.
* It has definition for AddSingleton, AddTransient, AddScoped.

### ServiceCollection**ContainerBuilder**Extensions class

* Includes BuiltServiceProvider extension method => which creates and return service-provider.

### Ways to get instance of IServiceProvider

#### Using IApplicationBuilder

* IApplicationBuilder.ApplicationServices property.

#### Using HttpContext

* HttpContext.RequestServices

#### Using IServiceCollection

* IServiceCollection.BuildServiceProvider

### IoC

* A very general principle.
* Flow of Control is “inverted” by dependency injection (a way to impl IoC) b/c u have efficiently delegated dependencies to some external system (like IoC container).

## A simple DI container

* Di container => typically wrap a dictionary with **System.Type as key** and **value will be some object that will allow to create instance of that type** (factory method).

## Resolving services inside the startup class

* Jo runtime ka hosting-provider hai, vho startup class ke constructor mai certain services inject kr skta hai. Services like, IConfiguration, IWebHostingEnvironment (IHostingEnvironment in pre 3.0 version), ILoggerFactory and IServiceProvider.
* The runtime’s hosting provider can inject certain services into constructor of the Startup class.
* Startup instance is built by hosting layer and contains only essential services for starting up an application.
* The ConfigureService() donot allow injecting services, yhe tho bs ek IServiceCollection as argument accept krta hai.
* Aur yhe baat sense bhi krti hai b/c ConfigureService() vho jagah hai jaha aap sirf apke application ko jo service cahiye vho register krte ho.
* Kher, u can use service injection in the startup’s constructor.
* Aur then, any services registered in ConfigureService() can be injected into Configure() method.
* i.e., u can add arbitrary no. parameters (services) after the IApplicationBuilder parameter.

## Manually resolving dependencies

* Agar apko manually service ko resolve krna hai, tho aap ApplicationServices property use krskte ho provided by IApplicationBuilder in **Configure() method**.
* Note: app Startup class ke constructor mai IServiceProvider as service pass krskte ho, but again it will contain limited subset of services => thus limited utility.
* Agar apko ConfigureServices() mthd mai services ko resolve krna hi hai, tho a different approach is required,
* Aap yhe kr skte ho ki, ek naya **IServiceProvider** ka instance bana lo from IServiceCollection instance jo contain kr rha hoga services jo iss point tk registered hui rhi hogi.
  + services.BuildServiceProvider();
* Note: Generally apko avoid krna cahiye to resolve service inside ConfigureServices() method.
* Kabhi kabhi apko bas Ioptions<MyOption> ke instance ko access krna rehta hai => jo ki aap krskte ho by binding the values of IConfiguration instance to an instance of Ioptions (which is essentially what the options fw does).
* Manually resolving services aka **Service Locator** is generally considered an anti-pattern. Fw or infrastructure layers ke liye iska use case hai, but u should avoid using it).