Unit Testing with IServiceProvider

When you create unit tests for class types that directly use DI to retrieve dependencies, you will need a way to give them a reference to a service provider (IServiceProvider).

This is transparently done by the runtime.
But, we have to mimic it for unit testing.

Here's how to make it happen.

ServiceProviderHelper

There's a class in OGA.Testing.Lib called, ServiceProviderHelper.
It includes a single call that will create and return a baseline service provider for you.
The baseline instance won't have anything in its DI registry, by default.
But, you can pass a callback to it, to register whatever your tests require.

To use it, you will need to compose a callback that registers any config and services that your test requires.
It will look something like this:

```
private void AddCustomServices(IServiceCollection services)
{
    // Firstly, register the root configuration instance with services, like the net core DI runtime does...
    services.AddScoped<IConfiguration>(_=> config);
    //services.AddScoped<IConfiguration, IConfigurationRoot>(_=> config);



    // Register the IOptions instance of our config...
    services.ConfigureWritable<BuildDataConfig>(cfgsection);



    // Register our mock service...
    // NOTE: We will let DI figure out constructor parms.
    services.AddSingleton<Service_Mock_A>();
    //services.Add(new ServiceDescriptor(typeof(InterfaceA), typeof(ClassA), ServiceLifetime.Singleton));
```

```
    // Register another service that also uses the same config...
    services.AddSingleton<Service_Mock_B>();
}
```

Then, you include in your unit test or test setup logic a call to Setup_DIProvider(), and give it the above callback.

Like this:

```
// Get a service provider with registered things for the test...
IServiceProvider sprov = ServiceProviderHelper.Setup_DIProvider(AddCustomServices);
```

Revision #1
Created 30 April 2025 21:11:29 by glwhite
Updated 30 April 2025 21:31:32 by glwhite