# ASP Core Configuration vererben

public IConfiguration Configuration { get; }

in ServiceCollection

services.Configure<BaseConfiguration>(this.Configuration);
services.Configure<ChildConfiguration>(this.Configuration);

Die ChildConfiguration erbt von BaseConfiguration

Hat die ChildConfiguration Unterknoten, erben auch die Unterknoten. So wird die Konfiguration „erweitert“

---

# Integrations Test mit Dependency Injection MsTest v2 .Net Core 3.1

Möchte man einen Integrationstest schreiben und dabei dependency injection nutzen, muss man folgendermaßen vorgehen:

1. Im Besten Fall teilt ma die Ausführende Applikation z.B. Web Applikation, Console etc und eine Bibliothek.
2. In die Bibliothek kommt die Auslagerung der StartUp: [crayon-6766c6eb9a1f2096680723/]
3. Im Testprojekt wird eine Basis Klasse definiert, die dieses Modul einliest. Nun kann in CofigureServices services.UseMyModule() aufgerufen werden oder der Host

selbst gebaut werden:
[crayon-6766c6eb9a1f7860463806/]

4. Die eigentliche Test Klasse erbt nun von diesen BaseUnitTest. Nun kann ein Propertiy erzeugt werden, welche einen Service aus der Dependency Injection ausliest:
[crayon-6766c6eb9a1f9237410992/]