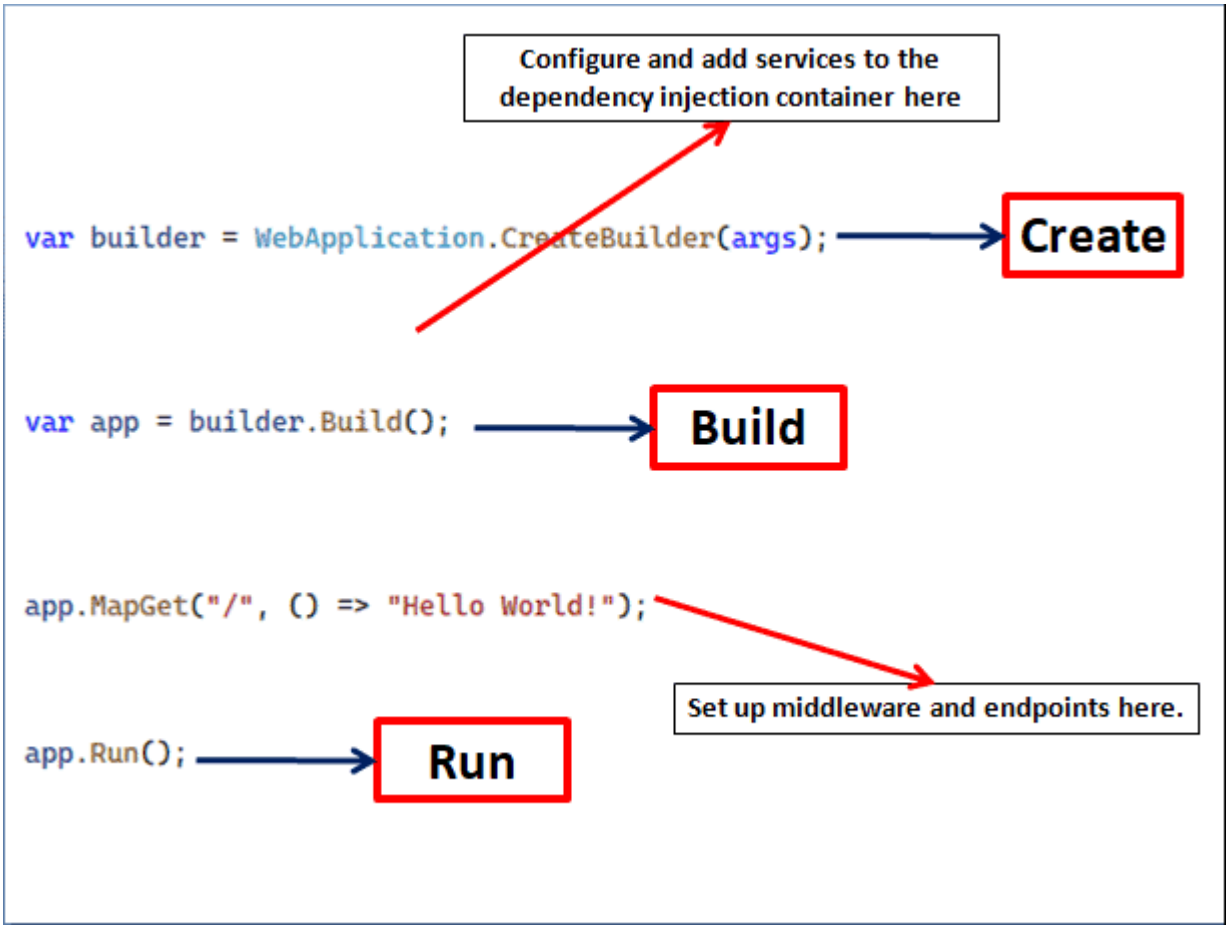
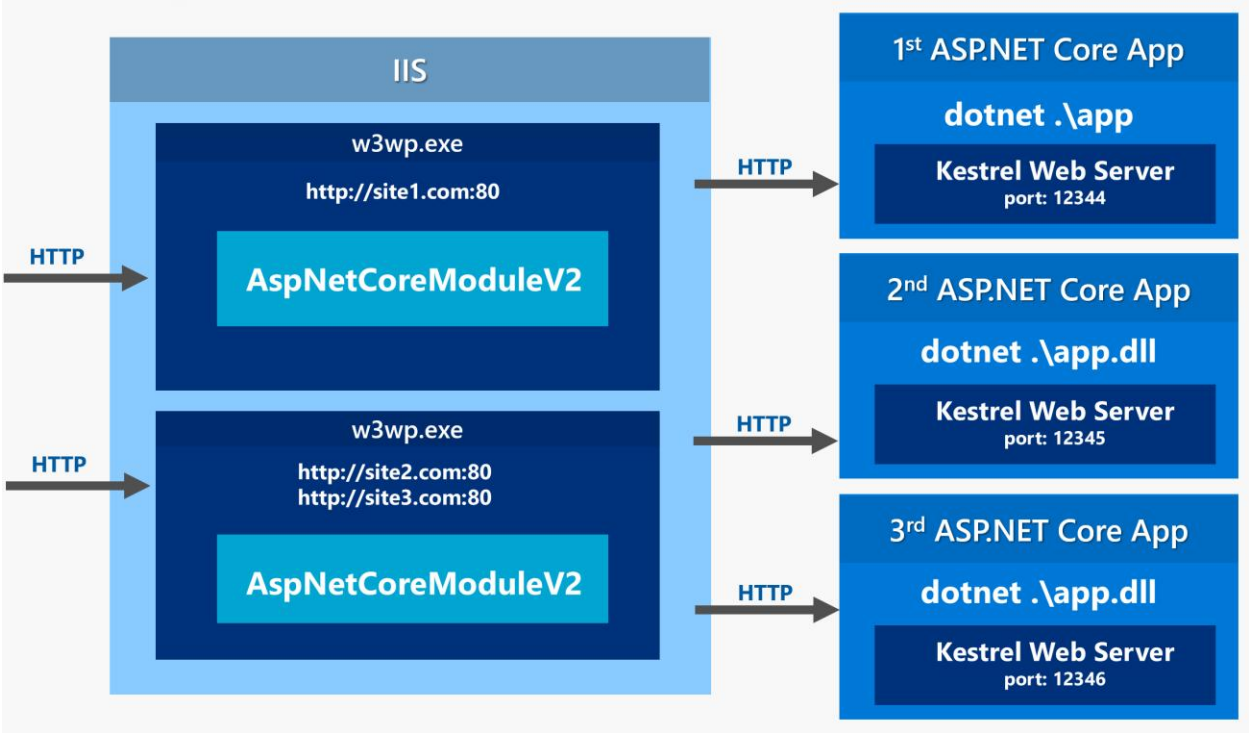
⭐ **Program.cs in ASP.NET Core**




IIS
w3wp.exe
http://site1.com:80
AspNetCoreModuleV2
w3wp.exe
http://site2.com:80
http://site3.com:80
AspNetCoreModuleV2
HTTP
HTTP
HTTP
HTTP
HTTP
1st ASP.NET Core App
dotnet .\app
Kestrel Web Server
port: 12344
2nd ASP.NET Core App
dotnet .\app.dll
Kestrel Web Server
port: 12345
3rd ASP.NET Core App
dotnet .\app.dll
Kestrel Web Server
port: 12346


ASP.NET Core 6.0
Minimal Hosting Model
SUBSCRIBE

In ASP.NET Core (especially .NET 6+), **Program.cs is the entry point of the application**.
It is the **brain of the project**, where the **application starts**, **services are registered**, and the **middleware pipeline is configured**.

Think of it like **Main() of the web application**.

🎯 1. What is Program.cs?

Program.cs contains the code that:

✓ Creates the web host
✓ Registers services (Dependency Injection)
✓ Configures middleware
✓ Runs the application

In .NET 6 and later, it uses the **Minimal Hosting Model** (no Startup.cs).

🎯 2. Program.cs Responsibilities

Here are the key responsibilities explained like an instructor:

✅ A. Application Startup / Entry Point

This is the first file that runs when the app starts.

var builder = WebApplication.CreateBuilder(args);

📌 **This line initializes the app**, loads configuration, sets up logging, etc.

✅ B. Register Services into DI Container

ASP.NET Core uses **Dependency Injection** everywhere.

Example:

builder.Services.AddControllers();

builder.Services.AddDbContext<AppDbContext>();

builder.Services.AddSwaggerGen();

📌 You register all your project services here:

- Controllers
- DbContext
- Repositories
- Identity
- Authentication & Authorization
- CORS
- Swagger
- Logging
- HttpClient
- AutoMapper

✅ **C. Build the Application**

```
var app = builder.Build();
```

📌 This compiles all the registered services and prepares the app to run.

✅ **D. Configure Middleware Pipeline**

Middleware decides:
“What happens with every HTTP request?”

Example:

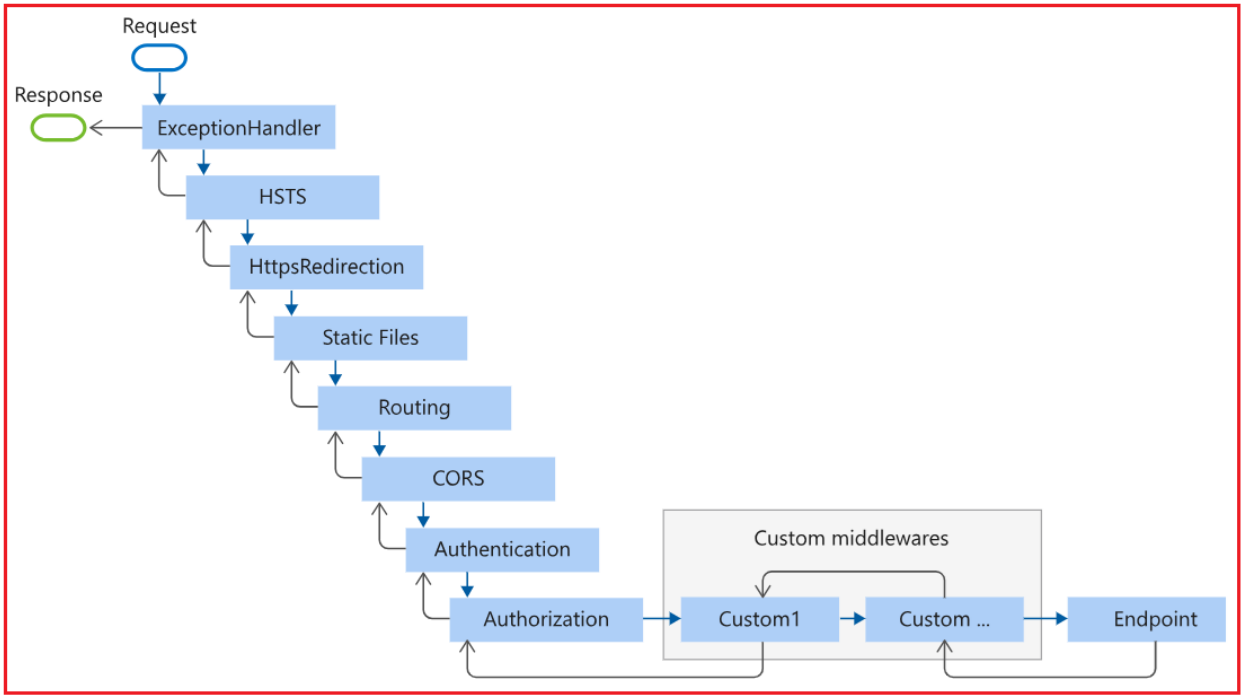
```
if (app.Environment.IsDevelopment())
{
   app.UseSwagger();
   app.UseSwaggerUI();
}
```

```
app.UseHttpsRedirection();
app.UseAuthentication();
app.UseAuthorization();
app.MapControllers();
```

Common middleware:

- UseHttpsRedirection
- UseRouting
- UseAuthentication
- UseAuthorization
- UseStaticFiles
- Custom Middleware
- Endpoints (MapControllers, MapGet, etc.)

❗ **Pipeline Flow Diagram**



🎯 3. Example Program.cs (Beginner Friendly)

```
var builder = WebApplication.CreateBuilder(args);

// 1. Register services
builder.Services.AddControllers();
builder.Services.AddEndpointsApiExplorer();
builder.Services.AddSwaggerGen();

// 2. Build app
var app = builder.Build();

// 3. Configure middleware
if (app.Environment.IsDevelopment())
{
```

```
    app.UseSwagger();
    app.UseSwaggerUI();
}

app.UseHttpsRedirection();
app.UseAuthorization();

app.MapControllers();

// 4. Run app
app.Run();
```

🎯 4. What Replaced Startup.cs?

Before .NET 6:

- Startup.cs had ConfigureServices() and Configure()
- Program.cs had only the host builder

Now:
➞ **Everything merged into Program.cs**

This makes the project simpler and cleaner.

🎯 5. Why Minimal Hosting Model?

Microsoft simplified the structure:

✓ Less boilerplate
✓ Faster setup
✓ Easier for beginners
✓ Unified place for configuration
✓ Works well for microservices (clean startup)

🎯 6. Instructor Summary (For Classroom)

Responsibility	Explanation
Entry Point	Application starts here
DI Configuration	Register services required for the app
Build Host	Prepare the application
Middleware Pipeline	Controls request/response flow
Endpoint Mapping	Map routes for controllers, minimal APIs, gRPC, etc.
Run Application	Start HTTP server (Kestrel)

🎯 7. One-Line Definition (Exam Answer)

Program.cs is the entry point of an ASP.NET Core application where the host is created, services are registered, middleware is configured, and the application pipeline is built and executed.