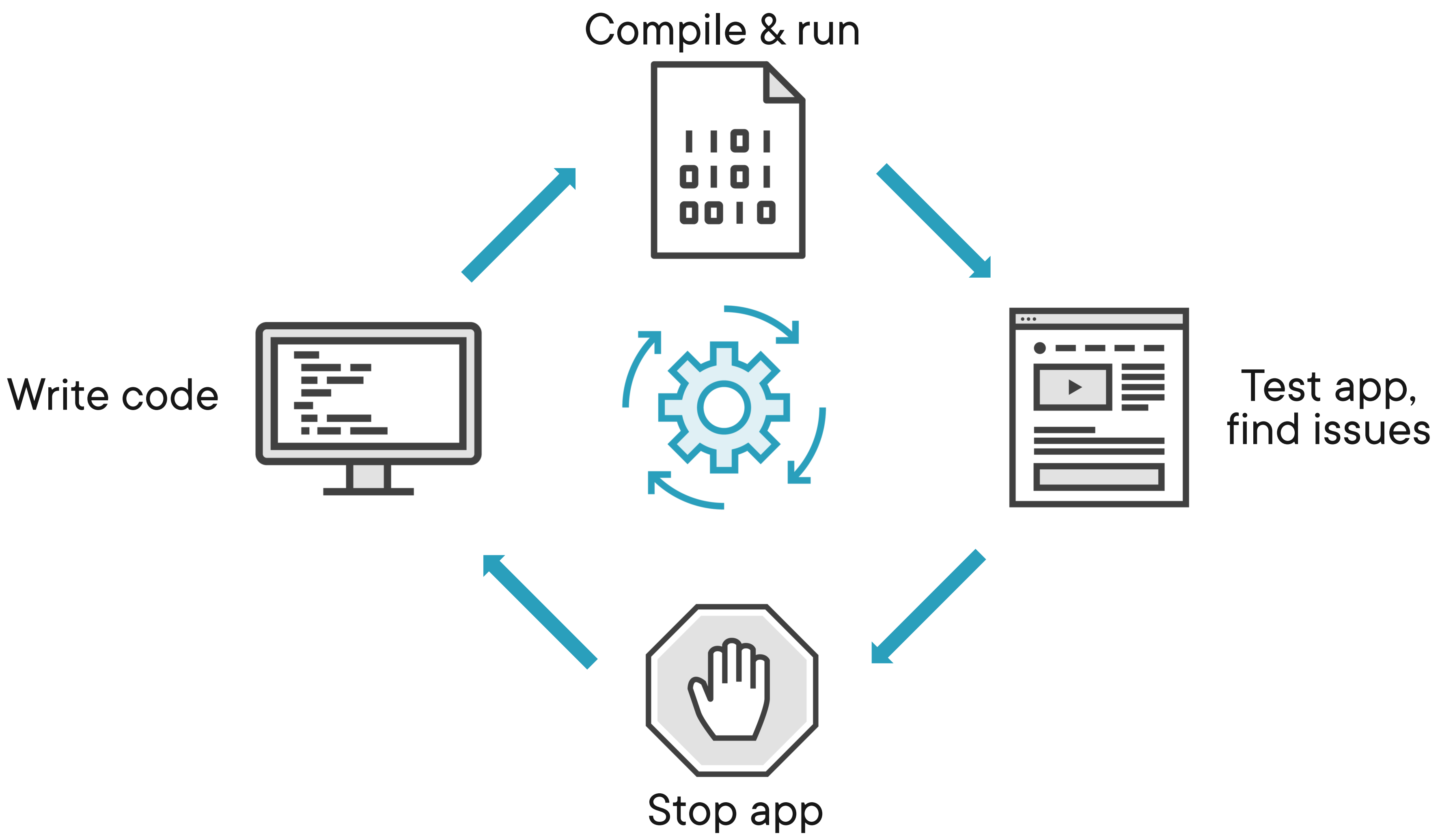
Improvements to MVC, Razor Pages, and ASP.NET Tooling

Alex Wolf
.NET Developer

www.thecodewolf.com

Introducing Hot Reload

.NET Development Workflow Challenges



Existing Options for Live Code Edits

Edit and Continue

Limited live changes while debugging

dotnet watch

Monitors the project for file changes and recompiles

Benefits of Hot Reload

Detects changes while the app is running

Applies updated assemblies to the current process

Maintains application state between changes

Visual Studio 2022 Tooling Features

- 64-bit performance
- Improved intellisense
- New syntax shortcuts
- Improved Git integration
- New ASP.NET project templates
- Updated SPA templates

Demo

Exploring Hot Reload with Visual Studio

Demo

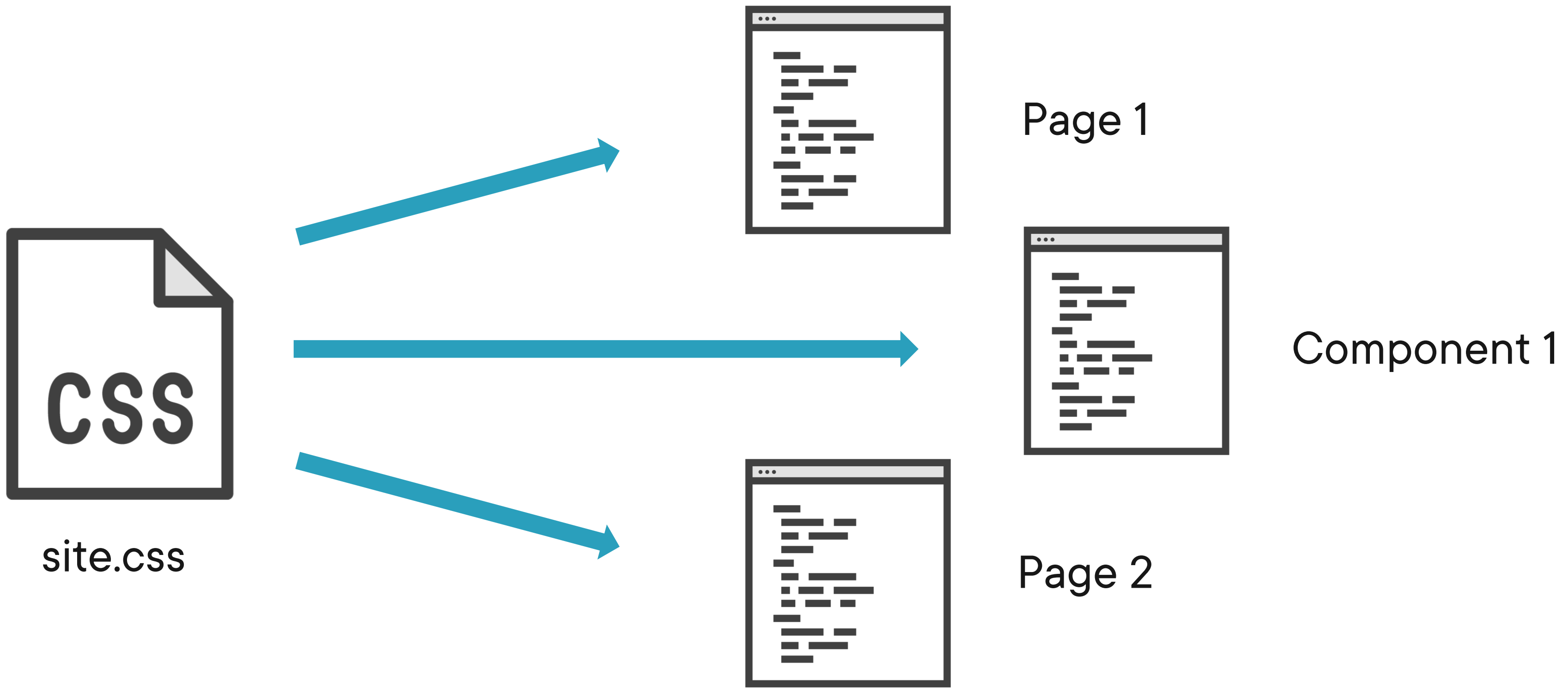
Exploring Hot Reload via the Command Line

Improvements to Razor Pages, MVC & Web API

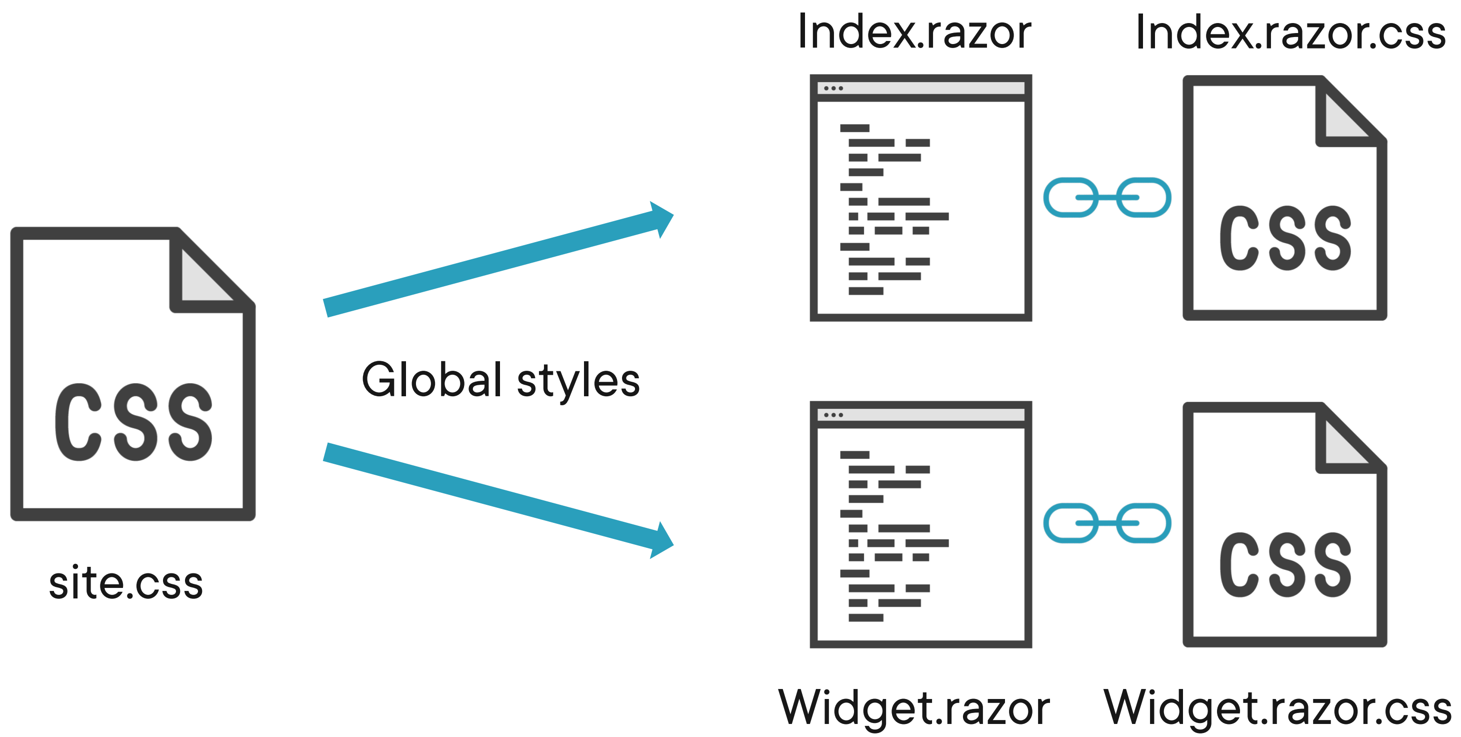
CSS Isolation

A CSS pattern that keeps styles scoped to a single page or component.

Traditional CSS Structure



Leveraging CSS Isolation



HTTP Logging in ASP.NET Core 6.0

Program.cs

```
app.UseHttpLogging();

builder.Services.AddHttpLogging(options =>

{

    options.LoggingFields = HttpLoggingFields.All;

    options.RequestHeaders.Add(“Custom Header");

    options.ResponseHeaders.Add(“Custom Header");

});
```

Http Logging Middleware

Logging configuration options using HttpLoggingOptions type

Miscellaneous Improvements

Razor Compiler

Improved internals & performance, unified assemblies

Razor Components

Now support optional parameters, improved async options

HTTP Headers

Simplified techniques for getting and setting HTTP Headers

Demo

Applying CSS isolation with Razor Pages

Demo

Improving HTTP logging

Understanding C# 10 Essential Features

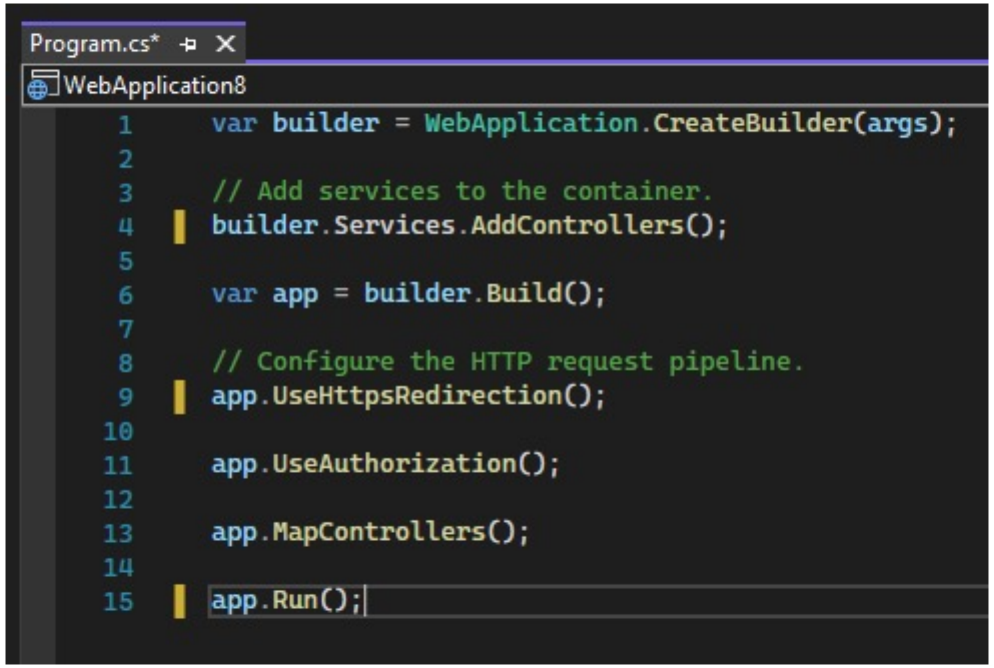
A Note About C#

Many new features added in recent versions

We’ll cover relevant essentials

Understanding Top Level Statements

Main()



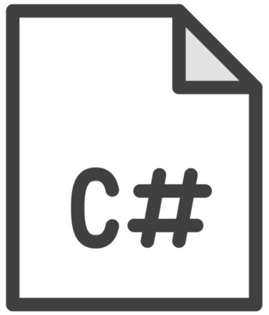
```
var builder = WebApplication.CreateBuilder(args);

// Add services to the container.
builder.Services.AddControllers();

var app = builder.Build();

// Configure the HTTP request pipeline.
app.UseHttpsRedirection();

app.UseAuthorization();

app.MapControllers();

app.Run();
```

Written at the root of the file

No visible namespaces or wrapper methods

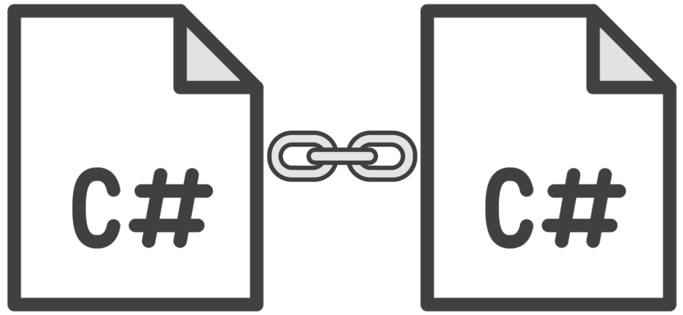
A Unified Program File

ASP.NET Core 6.0



Program.cs

ASP.NET Core 5.0



Program.cs

Startup.cs

(still works in 6.0)

Global Using Statements

Globals.cs

```
global using Microsoft.AspNetCore.Mvc;

global using Microsoft.EntityFrameworkCore;

global using Microsoft.EntityFrameworkCore.SqlServer;

global using System.Text;
```

Use the global keyword to apply a using statement to your entire project

Global usings can be centralized in one file, or distributed across many

Example Implicit Using Sets

Console app

```
System
System.Collections.Generic
System.IO
System.Linq
System.Net.Http
System.Threading
System.Threading.Tasks
```

Web app

```
System.Net.Http.Json
Microsoft.AspNetCore.Builder
Microsoft.AspNetCore.Hosting
Microsoft.AspNetCore.Http
Microsoft.AspNetCore.Routing
Microsoft.Extensions.Configuration
Microsoft.Extensions.DependencyInjection
Microsoft.Extensions.Hosting
Microsoft.Extensions.Logging
```

Remember, these features are all optional.

Other C# 10 features

Null parameter checks

File namespaces

Record structs

Lambda improvements

String interpolation enhancements

Many others!

Demo

Exploring Visual Studio project changes

Demo

Understanding SPA template improvements

Overview/ Summary

- Hot Reload allows us to make live edits to our running applications
- Hot Reload works in Visual Studio as well as via the command line
- Razor Pages and Views now support CSS isolation
- HTTP Logging can be used to easily view request traffic
- Getting and setting HTTP Request and Response headers is now simpler
- ASP.NET templates in Visual Studio utilize new platform and language features
- SPA templates have also been updated with more modern versions and patterns