Introduction to BAIT2113 Web Application Development

Week 1

Learning Outcomes

- Apply the concepts and issues in server-side Web Application
- Demonstrate skills in create, debug, optimize and maintain a secured Web Application with appropriate tools
- User server-side Web Application development to solve contemporary business problems

Assessment

Component	Contribution
Coursework	70%
Final Exam	30%

	Coursework Assessment	Due	Contribution
1	Assignment Part I	Week 4	32%
2	Assignment Part II	Week 7	48%
3	Mid Term Test	Week 4	20%

Assignment (3 - 4 members)

- Build a web site.

Part	Task	Due
Part 1	Program coding	Week 4 practical
Part 2	Program coding (extension from Part 1), presentation, and documentation.	Week 7 practical

You are advised to use the Microsoft Visual Studio 2022 to produce your work

Coursework mark

- *Make good is restricted.*
- *Students who* fail the Coursework will be required to REPEAT the course.

Introduction to Web Application

Chapter 1

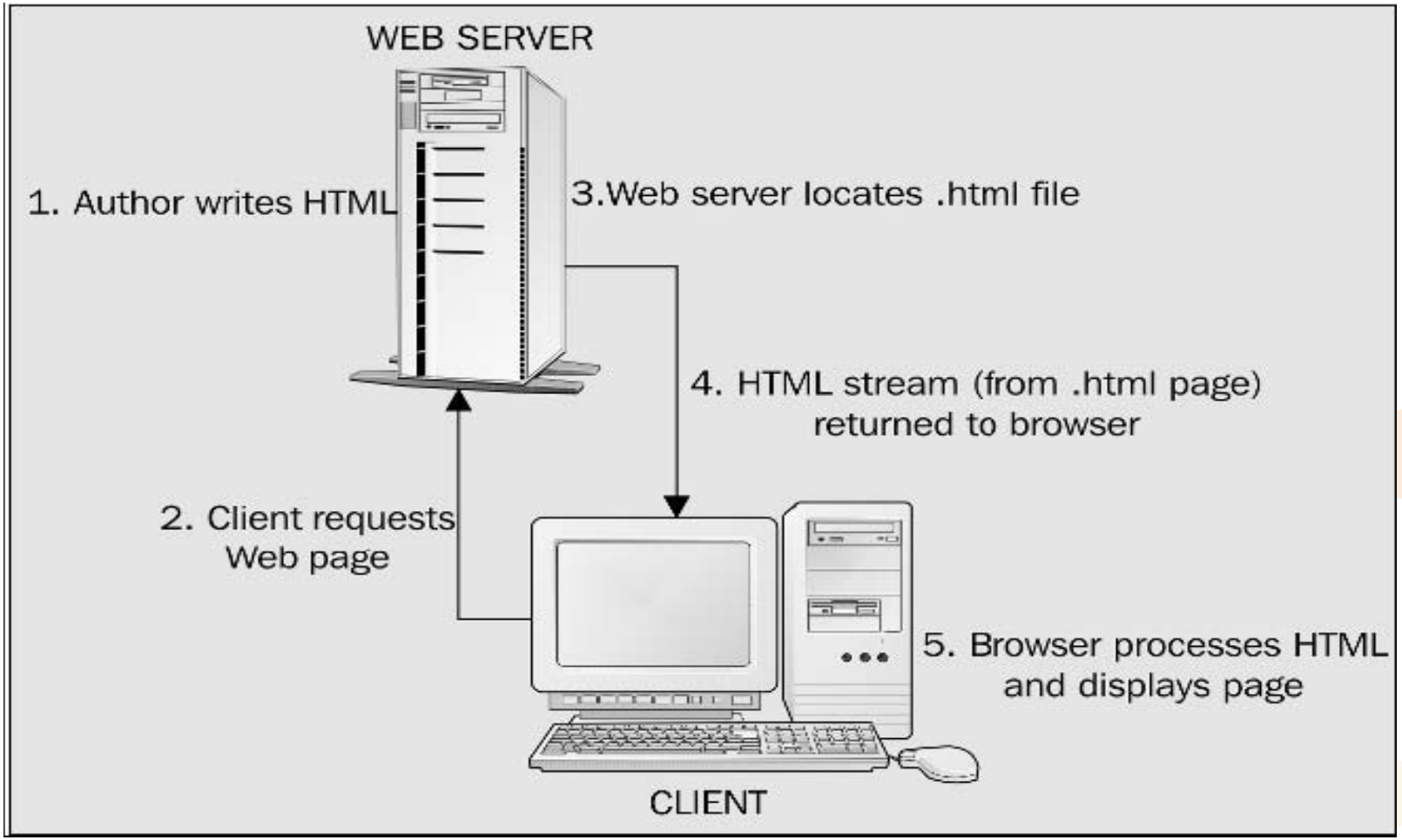
What Are You Going To Learn?

- At the end of this lesson, you will be able to:
  - Explain server-side
  - Differentiate between dynamic and static Web pages.
  - Explain the components in ASP.NET.
  - Study the .NET Framework.
  - Examine how ASP.NET works.
  - Create a simple ASP.NET page.

Static Web Pages

- The content (text, images, hyperlinks, and so on) and appearance of static Web pages is always the same.
- If you surf the Web, you'll see many static Web pages.

How Are Static Web Pages Served?



Static Web Page Example

```
<html>
<head>
    <title>Example of Static Page</title>
</head>
<body>
    <h2>Contact Me</h2>
    Tel: 03-41450123
</body>
</html>
```

What is the output if you run this code today and tomorrow?

Limitations of Static Web Pages

- Advantage:
  - FAST (as file size is small).
  - Lower cost.
- Disadvantage:
  - Limited/No functionalities.
  - Hard to maintain content.

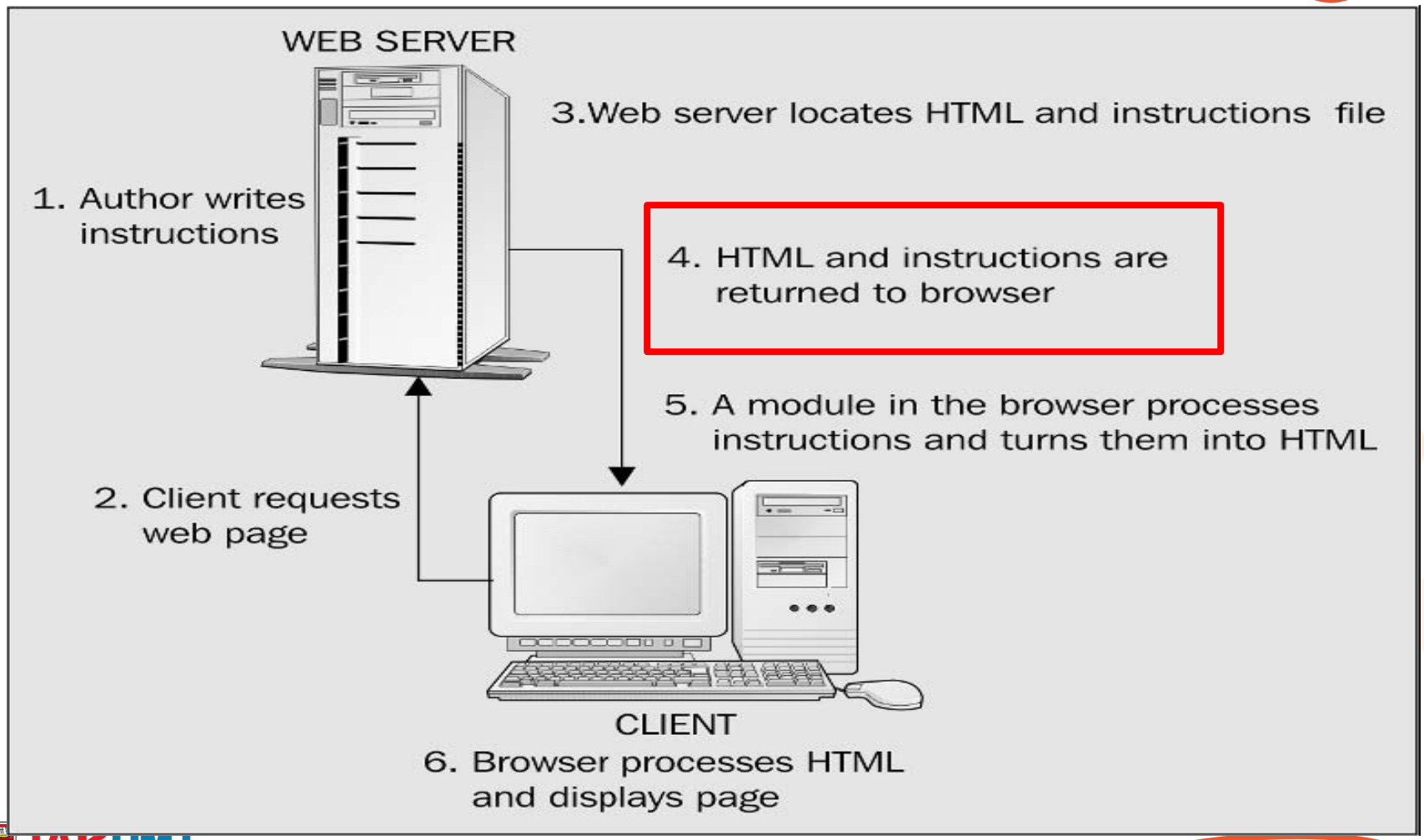
Dynamic Web Pages

- The appearance of dynamic Web pages is generated dynamically at run time.
- Thus the contents are not always the same.
- Examples of the dynamic Website:
  - News site
  - Web applications such as E-mail service.
  - Etc.

Dynamic Web Pages

- There are two ways in creating Dynamic Web Pages:
  - Client-Side Dynamic Web Pages
  - Server-Side Dynamic Web Pages

Client-Side Dynamic Web Pages



Client-side Dynamic Web Page Example

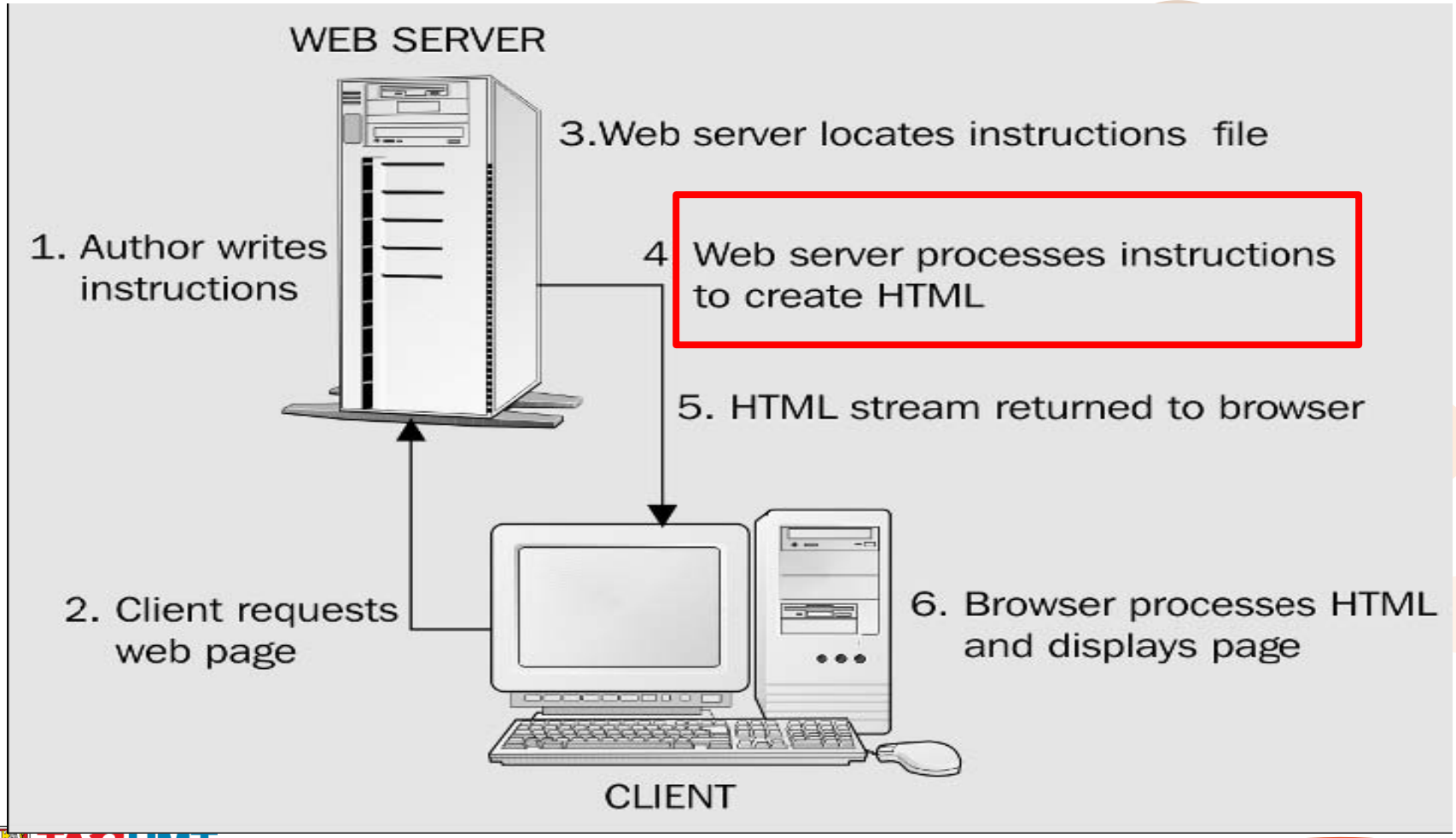
```
<!DOCTYPE html>
<html>
<head><title>Client-side Dynamic Web Page</title></head>
<body>

    <script>
    document.write(Date());
    </script>

</body>
</html>
```

What is the output if you run this code today and tomorrow?

Server-Side Dynamic Web Pages



Server-side Dynamic Web Page

```
<!DOCTYPE html>
<html>
<head><title>Client-side Dynamic Web Page</title></head>
<body>

    <p>The time is @DateTime.Now</p>

</body>
</html>
```

ASP.NET Razor

What is the output if you run this code today and tomorrow?

Differences between Client-side and Server-side Dynamic Pages

	Client-side	Server-side
Effects created by	Browser	Server
Language(s) used	Javascript	.NET (C#, VB.NET), PHP, Python, …
Response speed	Faster	Slower
Server memory consumption	Low	High
Limitations	Limited functionalities	More complex features such as database access, and web services.

Case studies

Consider the following problem, which technology will you use, client-side or server-side?

1. To show a confirm dialog box before the user deletes an item.
2. To change a button's image when user hovers the button.
3. To update the ticket balance after a user purchased a bus ticket online.
4. To display the status post to your online friends.

What is ASP.NET?

- ASP.NET is a server-side technology (not programming language) for creating dynamic Web pages.
- ASP.NET is a server-side technology that lets you use fully fledged programming languages to create your Web pages.

What is ASP.NET?

- ASP.NET pages can be made from one of many languages.
- ASP.NET is only one of a set of technologies that comprise the .NET Framework.
- Typical languages supported natively are C#, J#, VB.NET. On top of this, it is expected that third party developers will create versions of Perl, Python, and many others to work in ASP.NET.

.NET Framework

- A programming infrastructure created by Microsoft for building, deploying, and running applications and services that use .**NET** technologies, such as desktop applications and Web services.
- To use .NET, you must install the .NET redistribution package (which is also installed with Visual Studio).
- Evolved from version 1.0 to version 4.7 - as of this writing.
- For Linux, equivalent is the Mono open-source project.

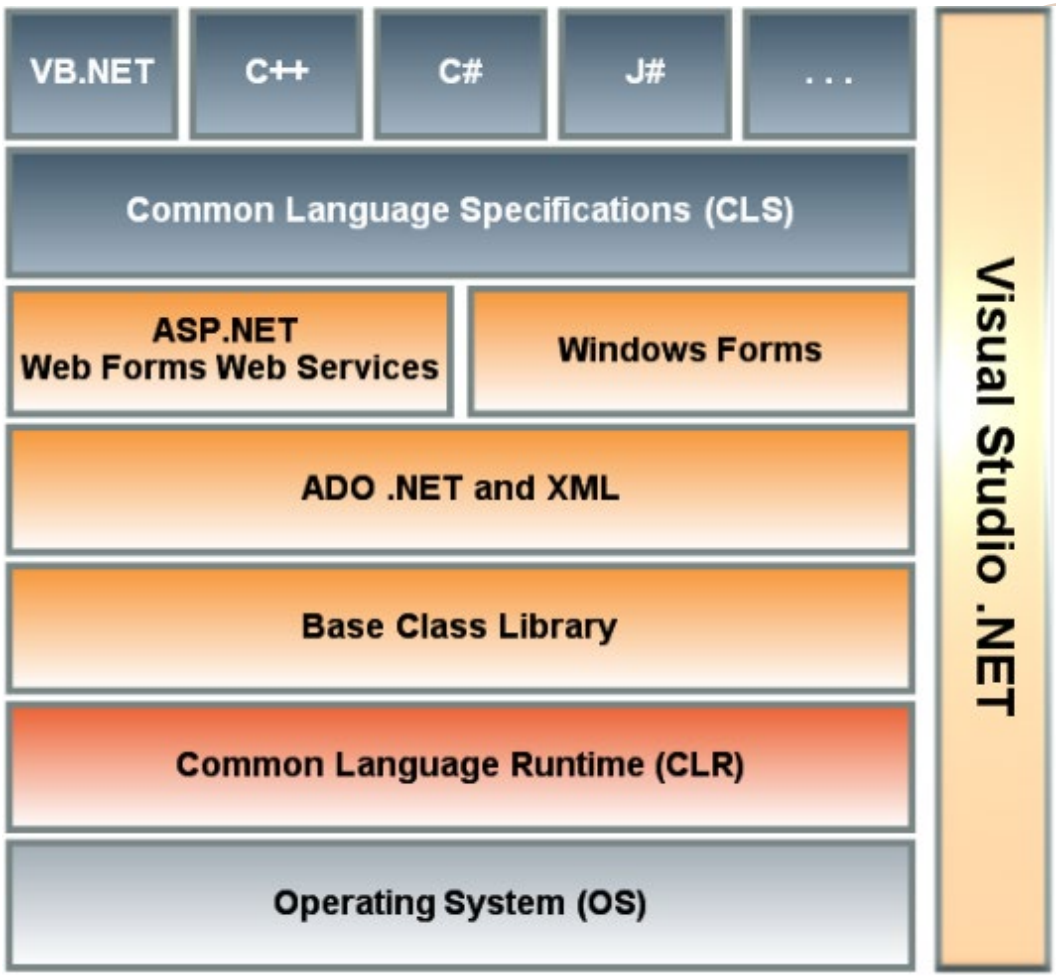
Introducing the .NET Framework

- The .NET Framework is an integral Windows component for building and running the next generation of software applications and Web services.
- The .NET Framework:
  - Supports over 60 different programming languages.
  - Enabling developers to focus on the core business logic code.
  - Makes it easier than ever before to build, deploy, and administer robust, and high-performing applications.
  - The .NET Framework is composed of the common language runtime and a unified set of class libraries.

Basic of .NET Framework

- We can break down our discussion of the entire .NET Framework into several core concepts:
  - Common Intermediate Language (CIL)
  - Common Language Runtime (CLR)
  - .NET Framework class libraries
  - .NET languages
  - Windows forms
  - ASP.NET
  - Web services

Features in .NET Framework



From Your Code to Machine Code

two types of compilation:

Pre-compiled code

- the code is compiled when we are done writing it and well before we need to use it

Interpreted code

- this code is compiled at the time of its execution (when the user requests the page)

Question

- What are the differences between pre-compiled code and interpreted code?

.NET Two-step Compilation

- .NET solves the problem by using a two-step process for compilation.
  - Common Intermediate Language (CIL)
  - Common Language Runtime (CLR)

.NET Two-step Compilation

- CIL and the CLR together give us the best of both worlds:
  - The structural optimization of pre-compiled code along with the portability of interpreted code.
- We can therefore use all compliant languages interchangeably within our applications – allows different teams to work on the same Web site in different languages.

Common Language Runtime (CLR)

- The CLR is responsible for executing your application code.
- When you write an application for the .NET Framework with a language such as VB .NET or C#, your source code is never compiled directly into machine code.

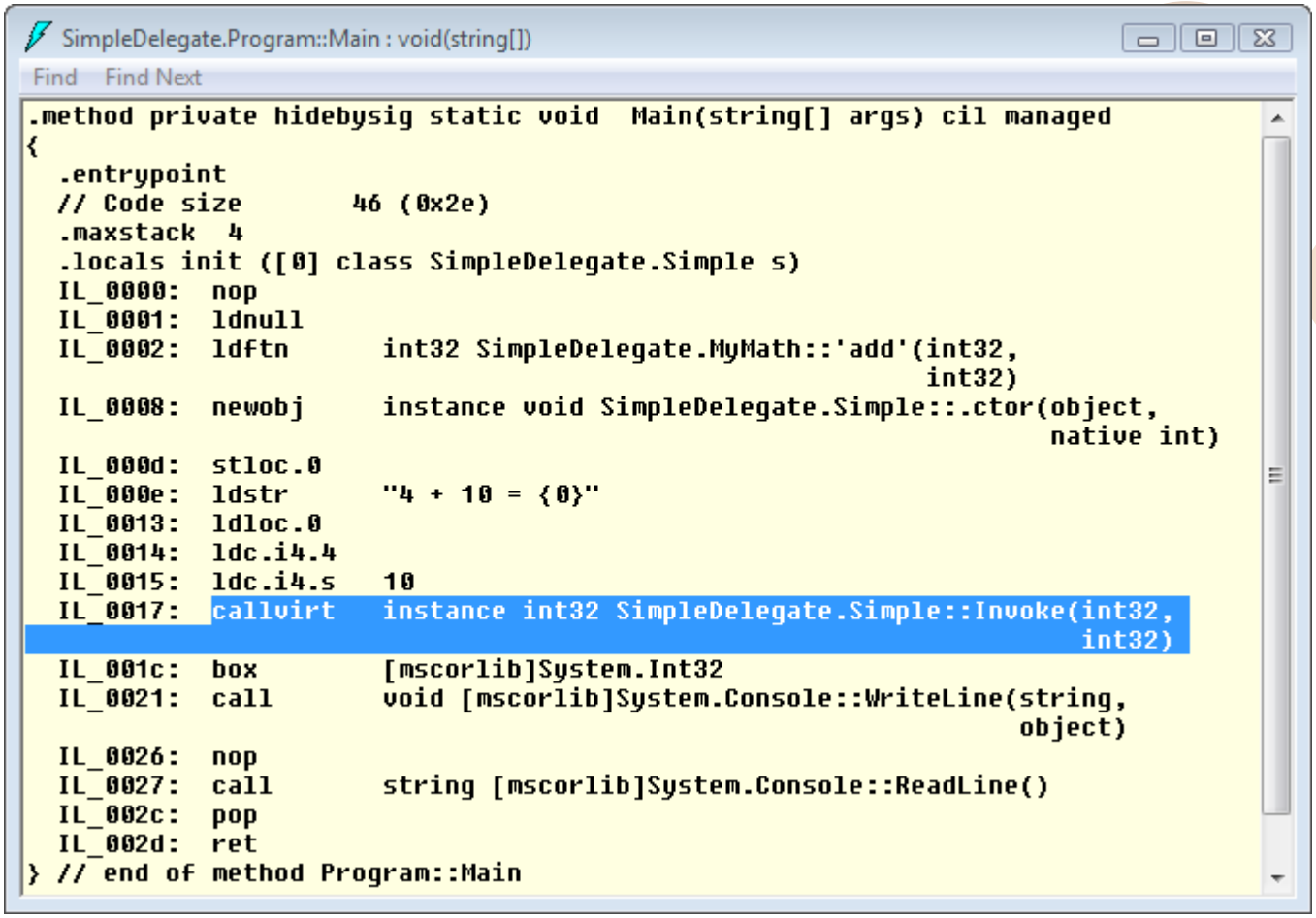
Common Language Runtime (CLR)

- Instead, the VB or C# compiler converts your code into a special language named:
  - CIL (Common Intermediate Language)

Common Language Runtime (CLR)

- CIL looks very much like an object-oriented assembly language. However, unlike a typical assembly language, it is not CPU specific.
- CIL is a low-level and platform-independent language.
- When your application actually executes, the CIL code is "just-in-time" compiled into machine code by the JITTER (the Just-In-Time compiler).

Common Language Runtime (CLR)

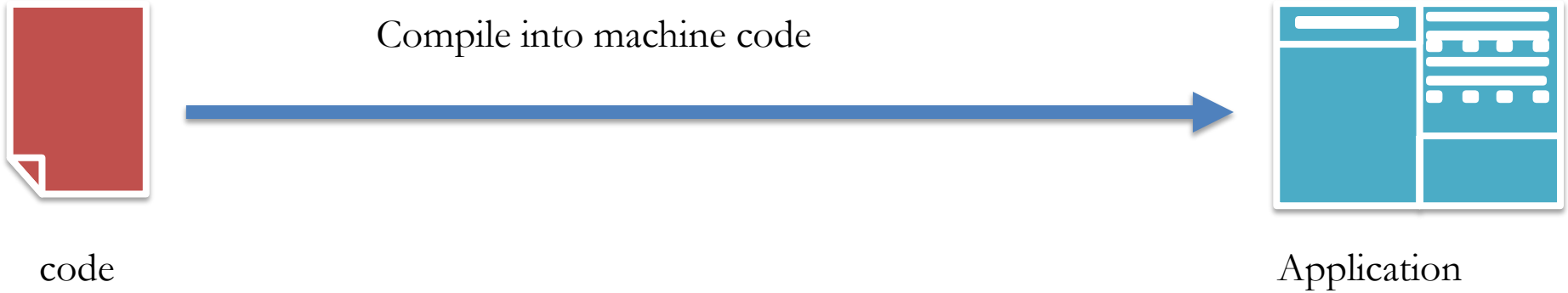


Common Language Runtime (CLR)

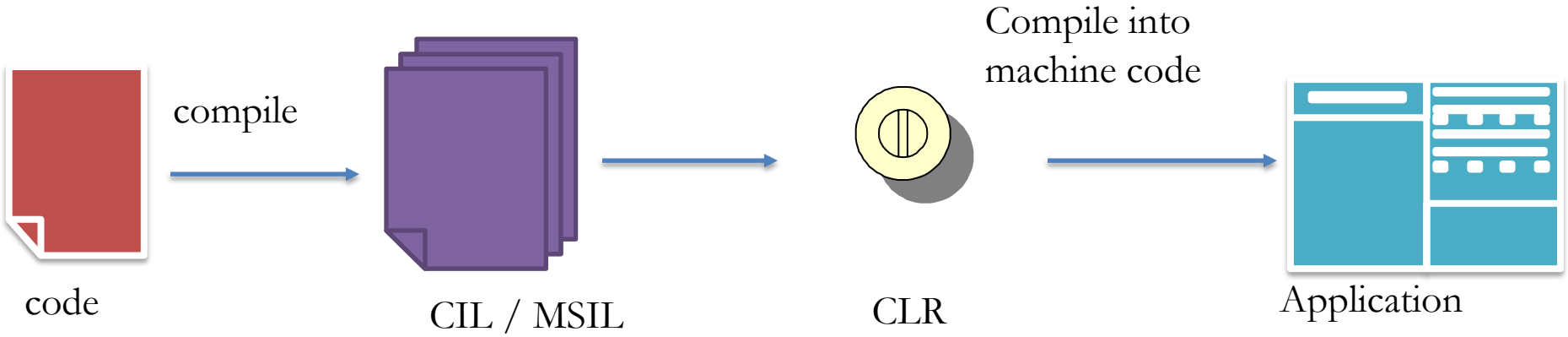
- The entire application is not compiled from CIL into machine code. Instead, only the methods that are actually called during execution are compiled.
- The .NET Framework understands only ONE language: CIL.
- However, you can write applications using languages such as Visual Basic .NET and C# for the .NET Framework

Common Language Runtime (CLR)

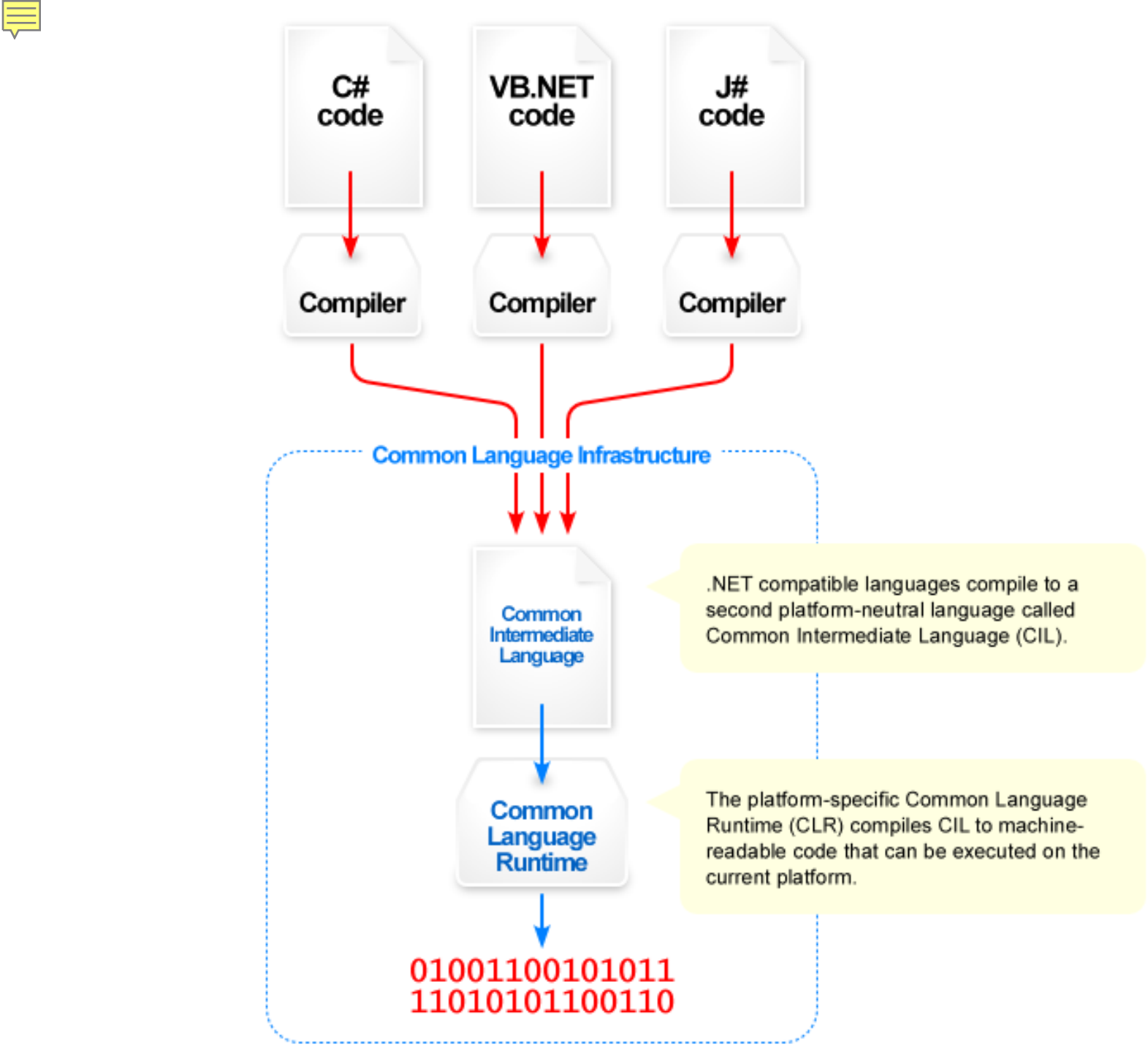
- Because the .NET Framework includes compilers for these languages that enable you to compile your code into CIL.
- You can write code for the .NET Framework using any one of dozens of different languages, including
  - Ada, Apl, Caml, COBOL, Eiffel, Forth, Fortran, JavaScript, Oberon, PERL, Pascal, PHP, Python, RPG, Scheme, Small Talk,etc.



Traditional Method (Compile to binary code)



In the .NET Framework


C# code
VB.NET code
J# code
Compiler
Compiler
Compiler
Common Language Infrastructure
Common Intermediate Language
.NET compatible languages compile to a second platform-neutral language called Common Intermediate Language (CIL).
Common Language Runtime
The platform-specific Common Language Runtime (CLR) compiles CIL to machine-readable code that can be executed on the current platform.
01001100101011
11010101100110

DEMO

Exploring VS2022

What Is a Web Server?

- Web servers are software that manage Web pages and make them available to client browsers – via a local network or over the Internet (two different machines or same machine).
- While there are many Web servers available (e.g. Apache, Internet Information Services (IIS))

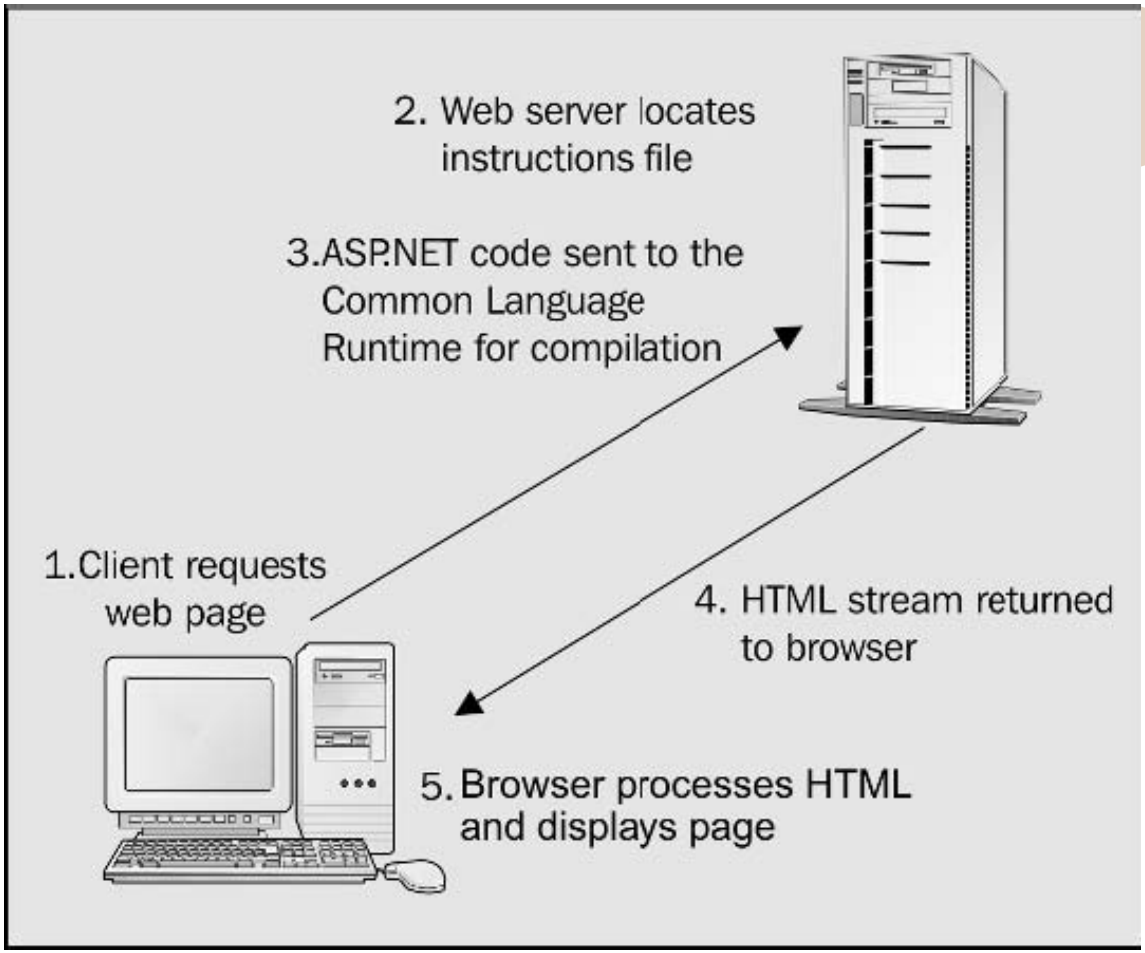
Hosting an ASP.NET Website

- To test or run ASP.NET Web applications, you need a Web server
  - Internet Information Services (IIS): built into Windows OS but not enabled by default.
  - Visual Studio Development Server: comes with Visual Studio
  - Third-party such as UltiDev Cassini Web Server.

How ASP.NET Works

- Think of ASP.NET pages as normal HTML pages that have sections marked up for special consideration.
- When .NET is installed, the local IIS Web server is automatically configured to look out for files with the extension ASPX and to use the ASP.NET module (a file called aspnet_isapi.dll) to handle them.

How ASP.NET Works



Creating First ASP.NET Web Page

- Simple example
  - Creating a simple ASP.NET page by saving your ASP.NET file with an ASPX suffix.

What Have You Learnt?

- Dynamic and Static Web Pages
  - Introducing the ASP.NET.
- Introducing the .NET Framework
- Exploring the Visual Studio 2022
- Installing a Web Server
- How ASP.NET Works
- Creating Our First ASP.NET Web Page