Date	Dec 11 2024	Unit	1

Introduction to .Net Technologies

Introduction to .NET Technologies

.NET is a comprehensive software framework developed by Microsoft. It provides a controlled environment for developing and running applications, including web, desktop, mobile, and IoT applications. The framework supports multiple programming languages like C#, VB.NET, and F#. With ASP.NET as its web component, developers can create dynamic web applications and services.

Introduction to Web Technologies

Web technologies are the foundational tools and techniques used to develop web applications. They encompass both client-side and server-side technologies. Here's an overview:

1. **Client-Side Technologies**:
    - Focus on user interaction and presentation.
    - Includes HTML, CSS, JavaScript, etc.
    - Executes on the user's browser.
2. **Server-Side Technologies**:
    - Focus on backend processing, database interaction, and business logic.
    - Includes .NET, PHP, Python, Java, etc.
    - Executes on the server.

HTML Basics

HTML (HyperText Markup Language) is the standard markup language for creating web pages. It structures web content and forms the backbone of web technologies.

Basic HTML Code

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width,
initial-scale=1.0">
    <title>Sample HTML Page</title>
</head>
<body>
    <h1>Welcome to .NET Web Technologies</h1>
    <p>This is a simple HTML page to demonstrate web technologies.</p>
    <a href="" target="_blank">Learn More About
.NET</a>https://www.microsoft.com/net
</body>
</html>
```

Client-Side Scripting

Client-side scripts run on the user's browser and enhance interactivity. Commonly used languages include JavaScript.

Advantages of Client-Side Scripts

- Reduces server load.
- Provides faster response times.
- Enhances user experience with interactive elements.

Disadvantages of Client-Side Scripts

- Browser-dependent behavior.
-
- Limited security due to visibility of code.

Sample Client-Side Script

```
<!DOCTYPE html>
<html lang="en">
```

```
<head>
    <title>Client-Side Script Example</title>
    <script>
        function showAlert() {
            alert("Hello! Welcome to .NET Technologies.");
        }
    </script>
</head>
<body>
    <h2>Client-Side Scripting Example</h2>
    <button onclick="showAlert()">Click Me</button>
</body>
</html>
```

Server-Side Scripting

Server-side scripts run on the server and handle data processing and logic. In .NET, ASP.NET is commonly used for server-side scripting.

Advantages of Server-Side Scripts

- Secure as the code is not exposed to the user.
- Handles complex tasks and interacts with databases effectively.

Disadvantages of Server-Side Scripts

- Increased server load.
- Slower response times compared to client-side scripts.

Sample Server-Side Script (ASP.NET Core Example)

1. **C# Controller (Backend Logic)**

```
using Microsoft.AspNetCore.Mvc;

namespace WebApp.Controllers
{
    public class HomeController : Controller
    {
        public IActionResult Index()
        {
```

```
            ViewData["Message"] = "Welcome to ASP.NET Core!";
            return View();
        }
    }
}
```

2. **Razor View (Frontend)**

```
<!DOCTYPE html>
<html lang="en">
<head>
    <title>ASP.NET Server-Side Example</title>
</head>
<body>
    <h1>@ViewData["Message"]</h1>
    <p>This message is generated server-side using ASP.NET Core.</p>
</body>
</html>
```

Comparison of Client-Side and Server-Side Tec

hnologies

Aspect	Client-Side	Server-Side
Execution Location	Browser	Server
Languages	HTML, CSS, JavaScript	C#, PHP, Python, Java
Security	Less secure	More secure
Dependency	Browser	Server
Use Cases	Interactivity, Validation	Data processing, Database

---

END