ITD523

Introduction to JavaScript fundamentals

Prepared by:

Assoc. Prof. Dr. Islam A.T.F. Taj-Eddin

Course Overview

- Basic programming concepts
- Development environments
- Your first JavaScript program: Hello, World!

What is Programming?

- Programming is communicating instructions to a computer
  - Programs are formal, unambiguous solutions to problems
  - Computers execute instructions faster and more accurately than humans
  - Programming languages bridge the gap between humans and machines

Programming Languages

- Thousands of programming languages exist
  - Languages differ by purpose and abstraction level
  - High-level languages are easier for humans
  - JavaScript is a high-level, general-purpose language

What is JavaScript?

- High-level, interpreted programming language
  - Originally developed in 1995 by Brendan Eich
  - Designed to add interactivity to web pages
  - Now used in web, server, mobile, and IoT applications

JavaScript as an Interpreted Language

- Executed by a JavaScript engine (interpreter)
  - Browsers include built-in JavaScript engines
  - Node.js allows JavaScript to run outside browsers
  - Uses Just-In-Time (JIT) compilation for performance

Client-Side vs Server-Side JavaScript

1. Client-side: runs in the browser
    - Used for interactivity and dynamic content
2. Server-side: runs on servers using Node.js
    - Used for backend logic and data processing

Where JavaScript is Used

- Web development (front-end and back-end)
- Mobile applications
- Web servers and APIs
- Games, IoT devices, and automation

Advantages of JavaScript

- Supported by all major browsers
- Large developer community
- Huge ecosystem of frameworks and libraries
- Easy to learn and use

Limitations of JavaScript

- Not suitable for high-performance computing
- Browser security restrictions (sandbox)
- Source code is visible to users
- Dynamic typing may lead to runtime errors

JavaScript Development Environments

- Online development environments
- Local development environments
- Browser-based execution
- Node.js-based execution

Online Development Environment

- No installation required
  - Used for learning and testing
  - Examples: JSFiddle, CodePen, JSBin, Plunker
  - Limited customization compared to local setup

Local Development Environment

- Closer to real-world software development
  - Highly customizable
  - Requires installation and configuration
  - Recommended for serious projects

Local Development Environment

- These will be, among others:
  - **package managers** –containing ready-made solutions that we can use in our programs, e.g. npm, yarn
  - **task runners and module bundlers** –automate the process of software development and merge the resulting code from many files and libraries, e.g. Grunt, Webpack
  - **testing framework** –automatic testing of the correctness of our program in search of potential errors , e.g. Mocha, Jasmine, Jest
  - **security analyzers** –to control the security of our solution,e.g. Snyk, RetireJS,OWASP Dependency Check)

Essential (minimal) Development Tools

– Code editor

– JavaScript interpreter

– Debugger

– Optional: package managers and testing tools

Popular Code Editors

– Visual Studio Code

– WebStorm

– Sublime Text

– Notepad++

JavaScript Interpreter

- Executes JavaScript code
- Browser engines (Chrome, Firefox)
- Node.js for server-side execution
- Checks syntax and runs code step by step

Debugger

- Used to analyze program execution
  - Pause and step through code
  - Inspect variables and program state
  - Built into modern web browsers

Developer Tools in Browsers

- Inspector for HTML elements
  - which will allow us, for example, to analyze the individual HTML elements of an open website;
- JavaScript console
  - which firstly shows all the information about the errors, and secondly allows us to run single JavaScript commands in the context of the current page;
- Debugger
  - which, among other things, shows the current values of variables, and allows you to pause code execution in the indicated place and to perform step-by-step work (i.e. execute single instructions of the program).
- Performance and network tools

Developer Tools in Browsers

– How do you enable the developer tools?

- Unfortunately, there is no single answer;

- Windows and Linux operating systems, all common browsers except Internet Explorer and Edge:

- Windows operating system, Internet Explorer and Edge:

- macOS operating system, all common browsers:

Your First JavaScript Program

- Traditionally called 'Hello, World!'
  - Used to test environment setup
  - Demonstrates basic syntax
  - Provides immediate feedback

Hello, World! Example

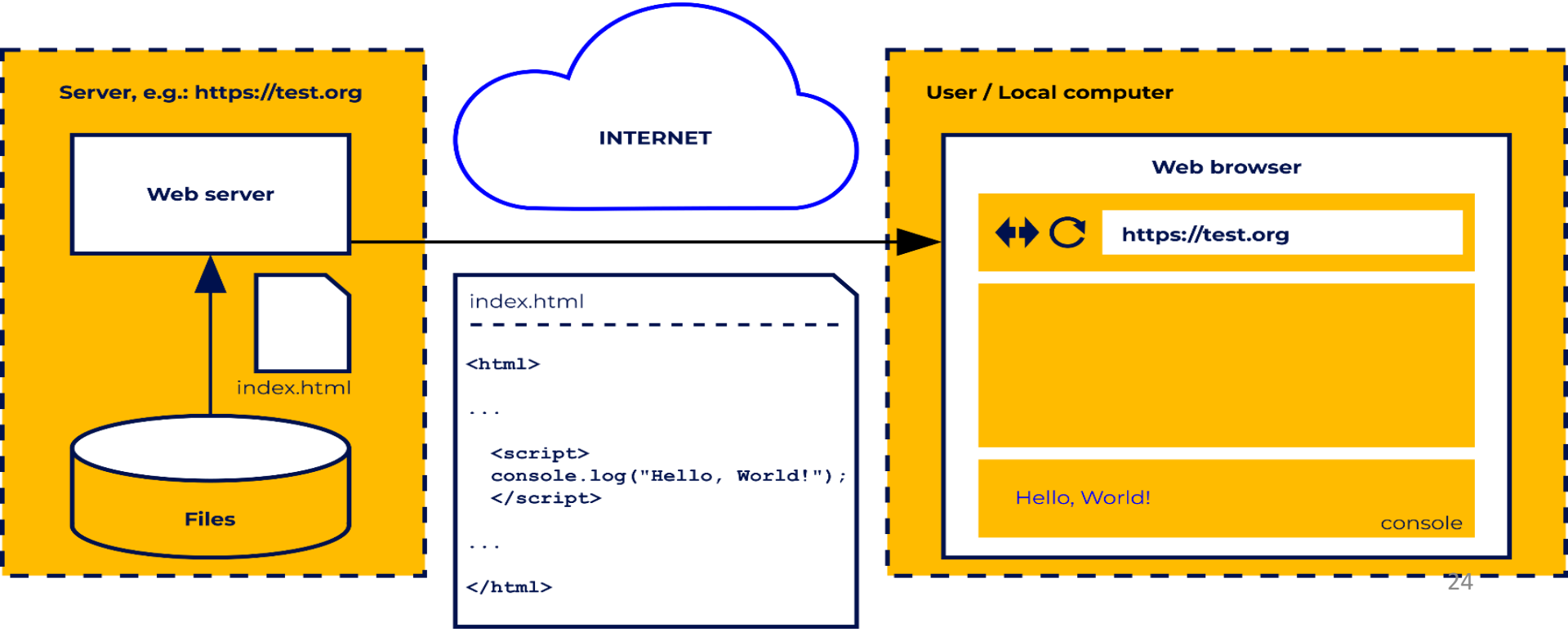
- console.log("Hello, World!");
  - console.log outputs text to the console
  - Semicolon ends the statement
  - Quotation marks define a string

Running JavaScript in the Console

- Open browser developer tools
  - Navigate to the Console tab
  - Type JavaScript commands directly
  - Useful for testing short code snippets

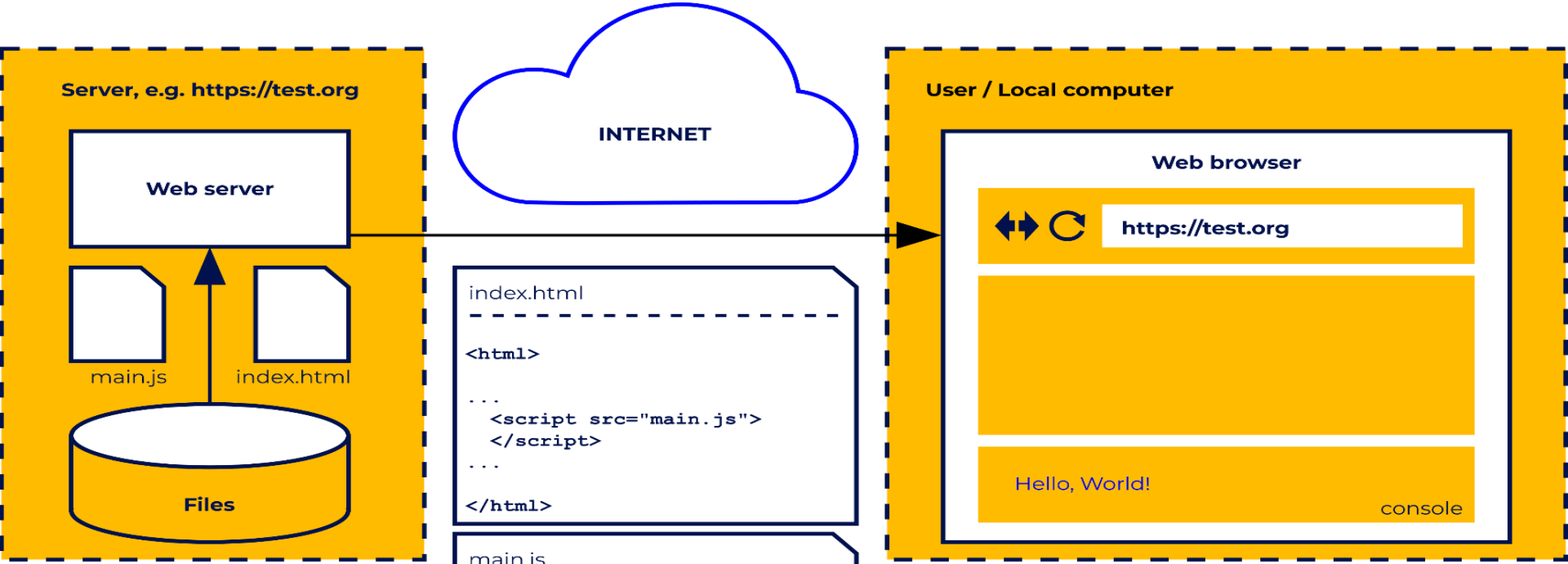
JavaScript with HTML

- JavaScript is embedded in HTML using <script> tag
  - Can be inline or external (.js file)
  - Executed when the browser loads the page
  - Forms the basis of web applications



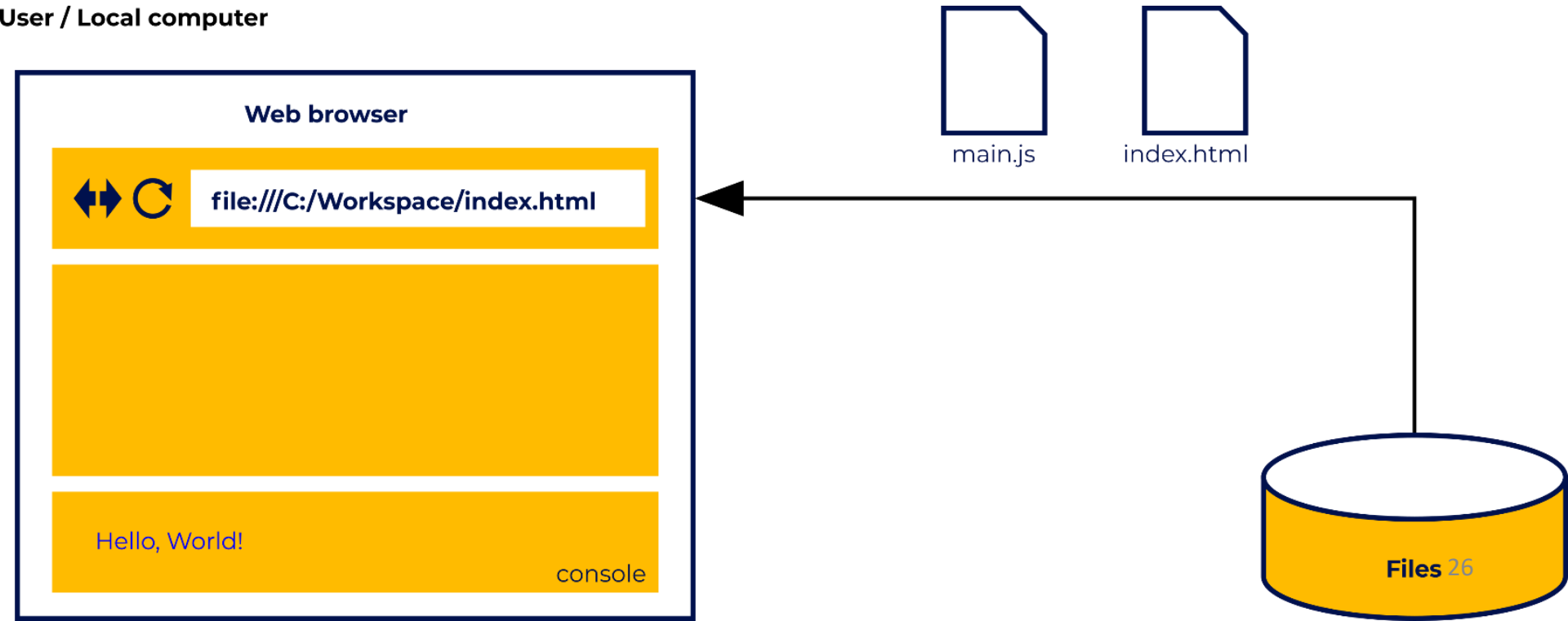
JavaScript with HTML

- JavaScript is embedded in HTML using <script> tag
  - Can be inline or external (.js file)
  - Executed when the browser loads the page
  - Forms the basis of web applications



JavaScript with HTML

- JavaScript is embedded in HTML using <script> tag
  - Can be inline or external (.js file)
  - Executed when the browser loads the page
  - Forms the basis of web applications



load a local html file

- By typing its local path after file:/// in the address bar
- By opening it in the browser using the Open command from the menu
  - Since the menu in browsers is very often hidden, a simpler way may be to use a shortcut to open existing documents in applications.

Windows

macOS

HTML, CSS, and JavaScript

- HTML: structure of the page
  - CSS: presentation and styling
  - JavaScript: behavior and interaction
  - Together form modern web applications

Learning Outcomes

- Understand basic programming concepts
  - Understand JavaScript fundamentals
  - Set up development environments
  - Run simple JavaScript programs

Summary

- JavaScript is a powerful and flexible language
  - Widely used in modern software development
  - Easy to start learning with basic tools
  - Foundation for future JavaScript modules