LESSON 36

HTML Lists

20 minutes | Introduction to Web Frontend | Outcome 1.22 | Outcome 3.3 | Outcome 2.20 | Outcome 2.21 | Outcome 1.16

WHAT THIS LESSON DOES

In this step-by-step lesson, you'll learn about HTML lists. Starting with an introduction to ordered and unordered lists, you'll explore how to code these lists and alter their appearance. You'll also delve into nested lists and have the opportunity to code your own list. Practical examples and interactive coding exercises are included to enhance your learning experience.

CURRICULUM OUTCOMES

Outcome	What the specification asks for
1.22	read, write, test, and modify computer programs
3.3	use appropriate programming languages to develop an interactive website that can display information from a database that meets a set of users' needs
2.20	identify and fix/debug warnings and errors in computer code and modify as required
2.21	critically reflect on and identify limitations in completed code and suggest possible improvements
1.16	compare two different user interfaces and identify different design decisions that shape the user experience

WHAT STUDENTS SHOULD BE ABLE TO DO

- Understand the purpose and structure of ordered and unordered lists in HTML.
- Master the use of HTML tags for creating ordered and unordered lists.
- Apply different numbering and bullet point styles using the type attribute.
- Create nested lists to organise complex information hierarchically.
- Develop practical skills by coding custom lists with varied content and styles.

BEFORE THE LESSON

- Open the in-browser code editor yourself and run one ordered and one unordered list so you know the Run Code button behaves and the preview renders.
- Confirm students can reach the lesson's code boxes on their machines; this lesson is worthless if the editor will not load behind the school filter.

HTML Lists

HOW THE LESSON RUNS

- Introduction
- Ordered Lists
- Different types of ordered lists
- Unordered Lists
- Different types of unordered lists
- Nested lists
- Code a list

TEACHING MOVES

- 1. Introduction. Draw the distinction concretely: ordered means the sequence matters (steps in a recipe, race positions), unordered means it does not (a shopping list). Ask for one example of each before showing any tags.
- 2. Ordered Lists. Stress the containment: li tags live inside ol. Have them type the lakes example exactly, run it, then add a sixth item themselves so they see the numbering update automatically.
- 3. Different types of ordered lists. Let them cycle the type value through a, A, i, I, 1 and run each. Note the source lists lowercase i and I both as lowercase Roman; point out I actually gives uppercase so they trust the output over the text.
- 4. Unordered Lists. Point out the only change from ordered is ol becoming ul; li is identical. Reinforce that the tag name, not the content, decides numbers versus bullets.
- 5. Different types of unordered lists. Have them try circle, disc and square and compare. Ask which they would pick for a website menu and why, tying styling to purpose.
- 6. Nested lists. This is the hard step. Show that the child list sits inside the parent li, not between li tags. Have them indent their code so the nesting is visible before they run it.
- 7. Code a list. Give a clear brief: at least five items, chosen topic, one nested sub-list, and a non-default type. Circulate and ask each student to point to their opening and closing ol or ul tag.

WHAT TO WATCH FOR

- Students place li tags outside the ol or ul, or forget the outer list tag and write only li items. Show that li alone renders unpredictably. Reinforce the rule: every li must sit between one ol or ul opening tag and its matching closing tag.
- In nested lists, students place the child list after the parent's closing li rather than inside the li. Highlight where the sub-list opens: before the parent li closes. Have them read their code aloud as 'Team A, and inside Team A these players'.
- Students assume a list that renders without errors is therefore correct, missing a stray unclosed tag. Ask them to check every opening tag has a matching close. A missing closing tag often still renders but breaks once more content follows.

DIFFERENTIATION

- Emerging: Give them a completed ordered list to copy and run first, then have them change only the content, so the tag structure stays fixed while they gain confidence.
- Developing: Ask them to convert an ordered list to unordered and back, explaining aloud which single change does it before they run each version.