HTML learning roadmap

Perfect question 👍 — because `z-index` is the CSS property most web developers hear about but don't always deeply understand in relation to **layers**.

Let's break it down:

🔹 What is `z-index`?

- In CSS, `z-index` tells the browser **which element should appear "in front of" or "behind" others** along the **Z-axis** (depth).
- Imagine the screen is a flat 2D plane (X = left-right, Y = up-down).
- The **Z-axis** is perpendicular to the screen (in-out).
  → `z-index` decides stacking order in that depth.

🔹 How it connects to Layers

- Every element is part of a **stacking context** (a sort of local 3D space).
- By default, elements paint in the order they appear in the DOM (later = on top).
- But if you apply `position` (relative, absolute, fixed, sticky) **and** `z-index`, the browser may create a **new layer** for that element.
- Then during **compositing**, the browser uses `z-index` to decide how those layers are stacked.

🔹 Example 1 — Simple Case

```html
<div style="width:200px; height:200px; background:red; position:relative; z-index:1;"></div>
<div style="width:200px; height:200px; background:blue; position:relative; z-index:2; margin-left:-150px;"></div>
```

- Red box has `z-index:1`.
- Blue box has `z-index:2`.
- Even though they overlap, **blue shows above red** because its `z-index` is higher.

🔹 Example 2 — Popup Modal

```css
.modal {
  position: fixed;
  z-index: 9999;
}
```

- The modal is drawn in a **new top layer** (fixed positioning).
- Huge `z-index` ensures it stays above everything else (navbar, content, footer).

That's why popups & dropdowns use **large** `z-index` **values**.

🔹 Key Developer Insights

1. `z-index` **doesn't always work**
   - If two elements are in different **stacking contexts**, one with lower `z-index` might still appear on top.
   - Example: parent with `opacity < 1` creates a new stacking context → its children can't escape it.
2. **High** `z-index` **everywhere is bad**
   - It can mess up compositing layers.
   - Better: manage stacking contexts carefully.
3. **Performance**
   - Elements with high `z-index` and transforms (`transform: translateZ(0)`) often create **GPU layers** for smooth animations.

✅ So in short:

- `z-index` **= depth order of layers (front/back).**
- Used during **compositing step** to decide which sheet lies on top.

Would you like me to **explain stacking context** next (the hidden reason why sometimes `z-index` "doesn't work")?