Chapter 3: Control Flow - Ready to Make Decisions in Your Code? 🚀

Control Flow

1. Conditional Statements (if, else if, else, switch)
2. Loops (for, while, do...while)
3. Nested Loops
4. Break and Continue Statements

Hey, LinkedIn Fam!

Ever wondered how apps know what action to take based on your input? Control flow is the key! In this chapter, we'll uncover how JavaScript **makes decisions** and **repeats tasks**, helping your code respond to different conditions, just like a "choose-your-own-adventure" story. From **conditional statements** to **loops**, you're about to learn how to control the flow of your code and bring logic to life!

1. Conditional Statements

Conditional statements are like "if-then" scenarios in real life:

- if: Runs code if a condition is true.
- else if: Checks additional conditions if the previous ones didn't match.
- else: Runs code if no other condition was met.
- switch: A structured way to check multiple values, especially useful when there are several possible conditions.

These statements guide your program to make decisions based on specific conditions.

2. Loops

Loops let you repeat code efficiently:

- for **loop**: Runs a set number of times—great for counting up or down!
- while **loop**: Continues until a condition is no longer true, useful for unpredictable scenarios.
- do...while **loop**: Runs at least once and then continues if the condition is true.

Loops save time and simplify your code by handling repetitive tasks.

3. Nested Loops

You can place one loop inside another—just like stacking gears in a machine! Nested loops are useful for tasks like processing arrays or working with multi-dimensional data.

4. Break and Continue Statements

- break: Stops a loop entirely when a certain condition is met.
- continue: Skips the current iteration and moves on to the next one.

These statements give you control over how and when loops should end or skip parts of their process.

Practice Questions:

1. **Create a** for **loop** that prints the numbers 1 to 10. If the number is 5, use break to stop the loop.
2. **Write a** switch **statement** that checks a day variable and prints "Weekend" for Saturday and Sunday, and "Weekday" otherwise.

🎁 Bonus Insights:

- **Infinite Loops**: Loops that never end can crash your program! Always make sure there’s a condition that will eventually stop them.
- **Choosing the Right Loop**: `for` loops are great for a known number of repetitions, while `while` loops work best when the condition might change mid-loop.

Stay Tuned:

Awesome job exploring control flow! In **Chapter 4**, we’ll take a closer look at **Functions**—the building blocks that let you package code into reusable units. Stay tuned for more as we level up your JavaScript journey!
