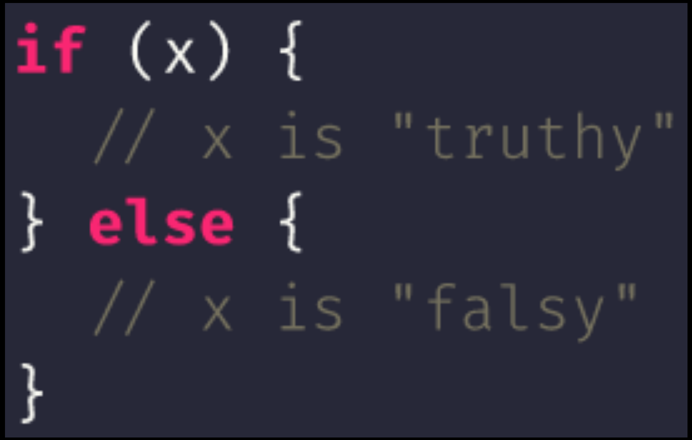
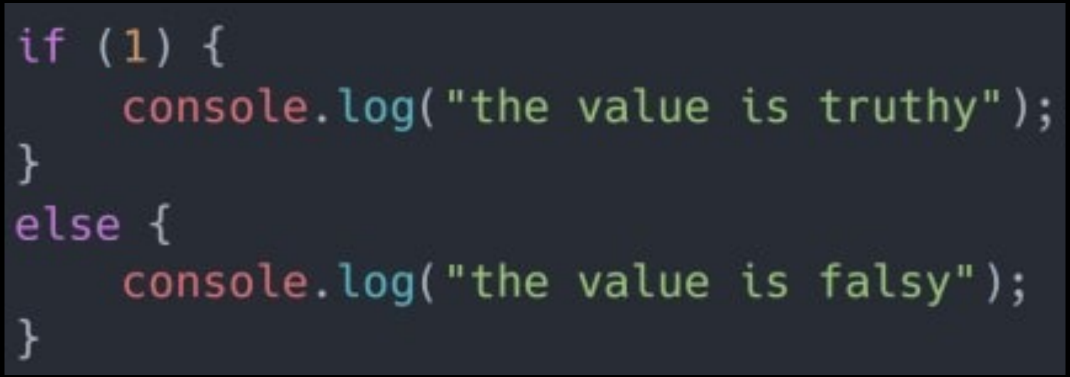
Truthy vs Falsy



```
if (x) {
  // x is "truthy"
} else {
  // x is "falsy"
}
```

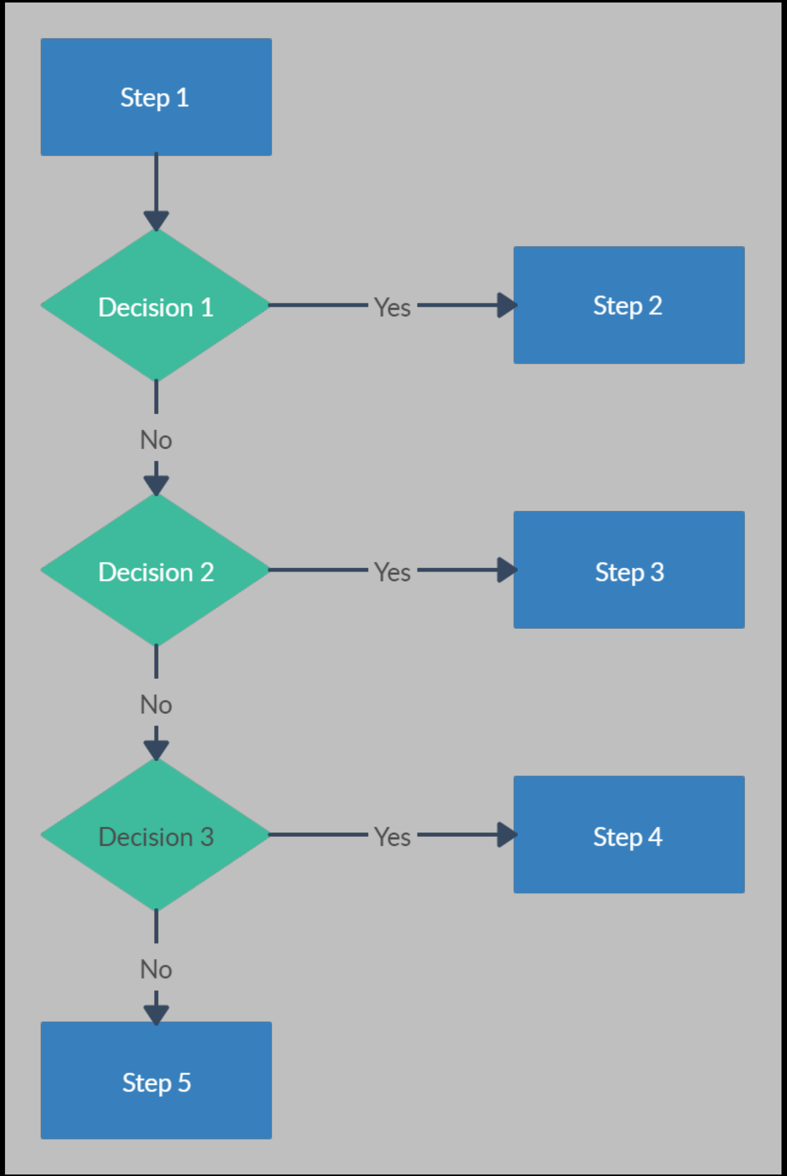


```
if (1) {
    console.log("the value is truthy");
}
else {
    console.log("the value is falsy");
}
```

1. Falsy Values: 0, null, undefined, false, NaN, "" (empty string)
2. Truthy Values: All values not listed as falsy.
3. Used in conditional statements like if.
4. Non-boolean values are auto-converted in logical operations.
5. Be explicit in comparisons to avoid unexpected behaviour.

if-else-if Ladder

1. Sequential Checking: The if-else ladder checks multiple conditions one after the other, from top to bottom.
2. First True Condition: Executes the block of code associated with the first true condition it encounters.
3. Exits After Execution: Once a true condition's code block is executed, it exits the ladder and skips the remaining conditions.
4. Fallback with Else: If none of the conditions are true, the final else block (if present) executes as a default case.

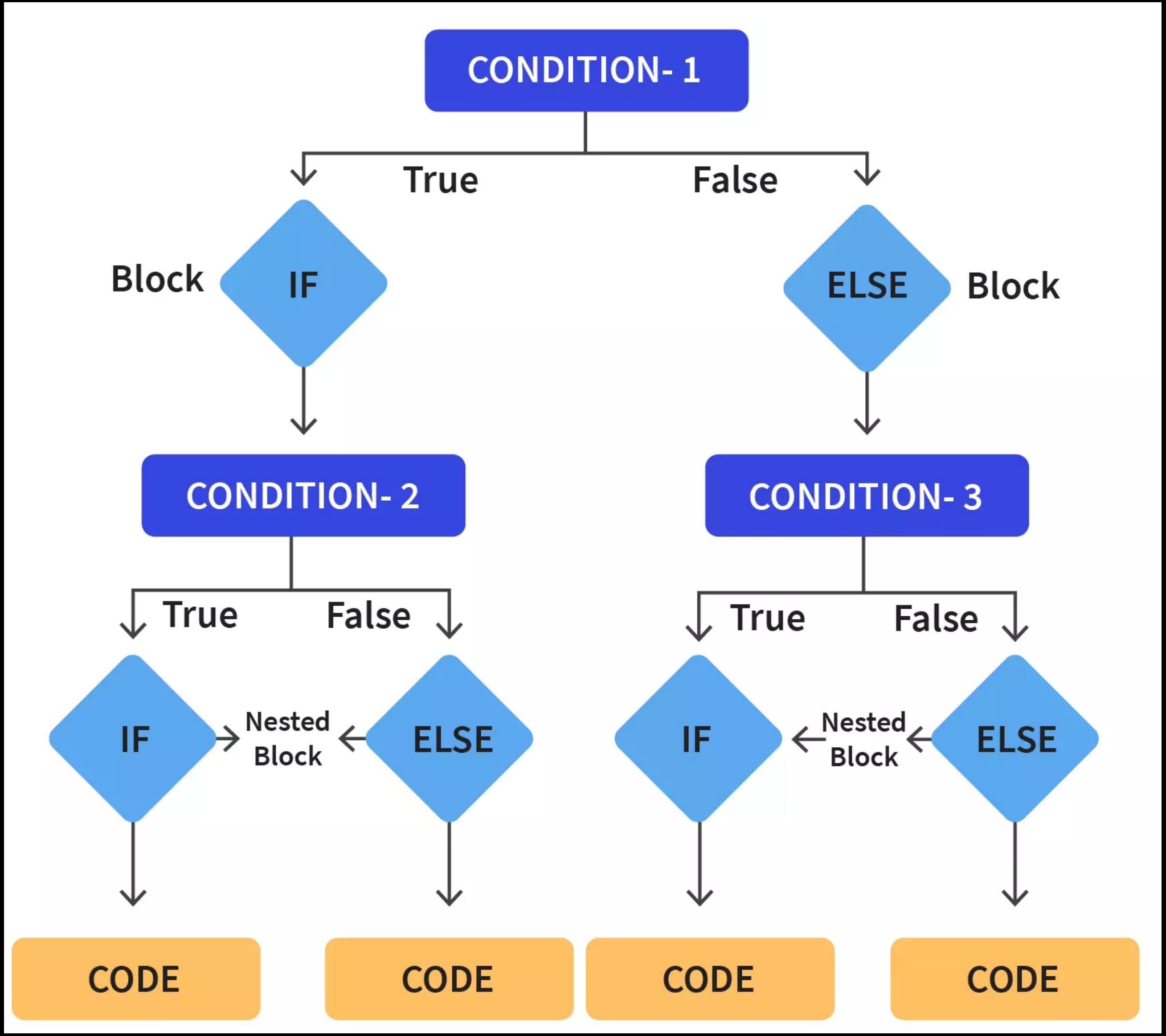


if-else-if Ladder

```
// Use of if-else ladder
let score = 85;
if (score >= 90) {
  console.log("Grade: A");
} else if (score >= 80) {
  console.log("Grade: B");
} else if (score >= 70) {
  console.log("Grade: C");
} else if (score >= 60) {
  console.log("Grade: D");
} else {
  console.log("Grade: F");
}
```

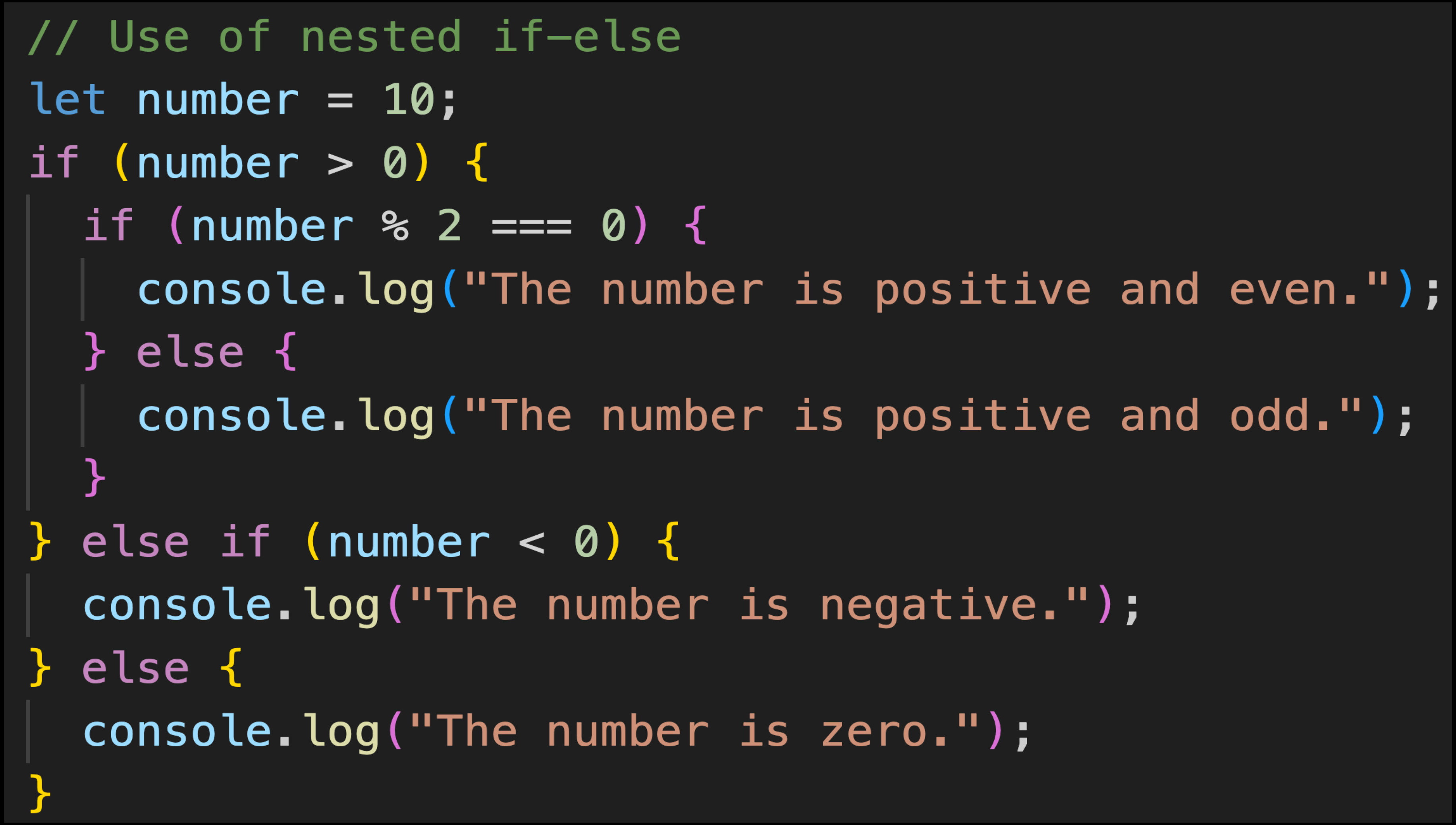
If-else Ladder: Multiple if and else if blocks; only one executes.

Nested if



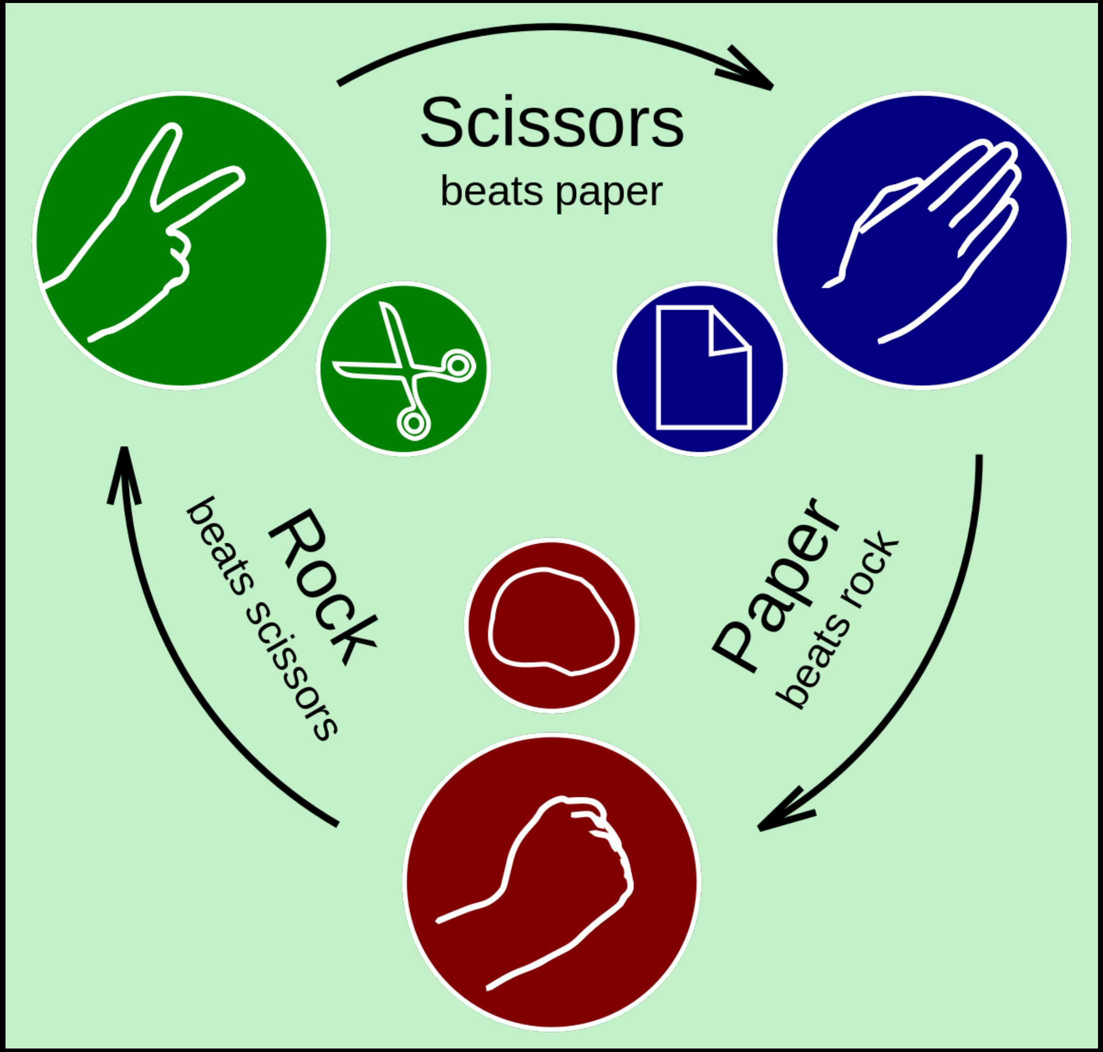
1. Condition Hierarchy: Enables hierarchical condition checks.
2. Complexity: Allows for detailed decision-making paths.
3. Syntax: An if inside another if or else.
4. Readability: Deep nesting can reduce code clarity.

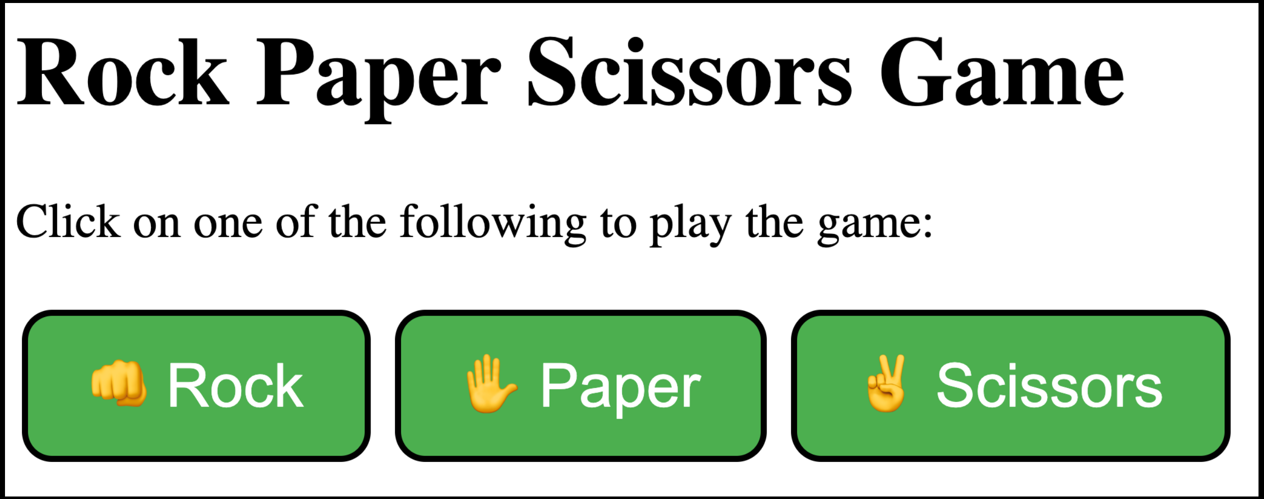
Nested if



```javascript
// Use of nested if-else
let number = 10;
if (number > 0) {
  if (number % 2 === 0) {
    console.log("The number is positive and even.");
  } else {
    console.log("The number is positive and odd.");
  }
} else if (number < 0) {
  console.log("The number is negative.");
} else {
  console.log("The number is zero.");
}
```

Project Rock-Paper-Scissor Game





Rock Paper Scissors Game

Click on one of the following to play the game:

- Create the UI
- Add click listeners to buttons to print the current choice

Logical Operators

AND

Or

NOT

1. Types: && (AND), || (OR), ! (NOT)
2. AND (&&): All conditions must be true for the result to be true.
3. OR (||): Only one condition must be true for the result to be true.
4. NOT (!): Inverts the Boolean value of a condition.
5. Lower Priority than Math and Comparison operators