



Java Learning Journey

Chapter 3: Selections in Java

 **Prepared by:** Ahmed Elsifi

◇ Boolean Data Type

- The `boolean` data type can hold only two values: `true` or `false`.
- Example:

```
boolean isValid = true;
boolean isEven = (number % 2 == 0);
```

- Relational operators (`<`, `<=`, `>`, `>=`, `==`, `!=`) return boolean values.
-

◇ Selection Statements

1. One-Way `if` Statement

```
if (condition) {
    // code to execute if condition is true
}
```

2. Two-Way `if-else` Statement

```
if (condition) {
    // true case
} else {
    // false case
}
```

3. Nested and Multi-Way `if-else`

```
if (score >= 90) {
    System.out.println("A");
} else if (score >= 80) {
    System.out.println("B");
} else {
    System.out.println("F");
}
```

4. `switch` Statement

- Used for multiple conditions based on a single value.
- Must use `break` to avoid fall-through behavior.
- Works with `char`, `byte`, `short`, `int`, `String`.
- Example:

```
switch (day) {  
    case 1: System.out.println("Monday"); break;  
    case 2: System.out.println("Tuesday"); break;  
    default: System.out.println("Invalid");  
}
```

◇ Logical Operators

- `&&` (AND), `||` (OR), `!` (NOT), `^` (XOR)
- Short-circuit evaluation: `&&` and `||` skip evaluating the right operand if the result is already determined.

◇ Conditional Operator (Ternary Operator)

```
result = (condition) ? valueIfTrue : valueIfFalse;
```

Example:

```
String message = (score >= 60) ? "Pass" : "Fail";
```

◇ Generating Random Numbers

Using `System.currentTimeMillis()`

```
int num1 = (int)(System.currentTimeMillis() % 10); // last digit  
int num2 = (int)(System.currentTimeMillis() / 10 % 10); // second last digit
```

Using `Math.random()`

```
int randomNum = (int)(Math.random() * 10); // 0 to 9
```

◇ Common Errors and Pitfalls

- Forgetting braces `{}`
 - Using `=` instead of `==`
 - Dangling `else` ambiguity
 - Testing equality of floating-point numbers (use epsilon tolerance)
 - Redundant boolean comparisons
-

◇ `System.exit(status)`

- Terminates the program.
 - `status = 0`: normal termination.
 - `status != 0`: abnormal termination (e.g., error).
 - **Not the same as `return 0`; in C:**
 - `return` exits the current method.
 - `System.exit()` terminates the entire JVM.
-

◇ Operator Precedence and Associativity

- Parentheses `()` have the highest precedence.
 - Logical operators have lower precedence than relational and arithmetic operators.
 - Most binary operators are left-associative; assignment operators are right-associative.
-

◇ Debugging Tips

- Use print statements to trace values.
- Use debuggers (e.g., Eclipse, IntelliJ) to:
 - Step through code
 - Set breakpoints
 - Inspect variables