Selection

Making decisions

Lesson Objectives

- Understand selection control logic
- Learn to use JavaScript control logic syntaxes

Control logic

- Sequence
- **Selection**
- Repetition

Use of relational and logical operators

- As in real life, we need to make decisions in our programs too.
  - we need to perform different actions based on different conditions.

- In programs, need operators that evaluate expressions to true or false

The “if” statement

- Selective execution can be done with the use of `if(...)` statement.
  - The `if(...)` statement evaluates a condition in parentheses and, if the result is true, executes a block of code.

```
if(condition){
// statements;
}
```

- See example: *lecture_codes/lesson3/selection_statements_if.js*

recommended to wrap your code with curly braces {…}, even if there is only one statement to execute.

The “else” clause

- `if` statement may contain an optional “`else`” block. It executes when the condition is false.

```
if(condition){
// statements;
}else{
// different statements;
}
```

- See example: *lecture_codes/lesson3/selection_statements_if_else.js*

Exercise

- Write a program that asks user to enter weather for today and print "Get an umbrella" if weather is “rainy”.

- Write a program that asks user to enter a number between 1 to 10, and print "Bingo!" if user enters 7 otherwise prints "Try again.".
  - Use === for comparison not ==

Several conditions: “else if”

- Sometimes, we’d like to test several variants of a condition.
  - The `else if` clause lets us do that.

```
let year = prompt('In which year was ES6 released?', '');

if (year < 2015) {
  alert( 'Too early...' );
} else if (year > 2015) {
  alert( 'Too late' );
} else {
  alert( 'Exactly!' );
}
```

  - There can be many `else if` blocks. The final `else` is optional.
- *See example: lecture_codes/lesson3/selection_statements_else_if*

Exercise

- Write a program that accepts user age as input and output following based on the input
  - If age <= 0
    - print "please enter valid age"
  - if age is between 0 and 14
    - print "You can't drive yet."
  - if age is between 15 and 18
    - print "You can drive under supervision."
  - if age is 19 or higher
    - print "You can drive."

Conditional (ternary) operator

- The conditional (ternary) operator is the only JavaScript operator that takes three operands: a condition followed by a question mark (`?`), then an expression to execute if the condition is truthy followed by a colon (`:`), and finally the expression to execute if the condition is falsy. *This operator is frequently used as a shortcut for the if statement*.
    - The value of the evaluated expression is returned
- *See example: lecture_codes/lesson3/ternary_operator*
- Only use this for very simple conditions
    - Probably best to avoid until become experienced and comfortable with standard if else

Switch

- A switch statement can replace multiple if (or else if) checks
- The `switch` has one or more `case` blocks and an optional `default`.

- Using else if – See example: *lecture_codes/lesson3/using_else_if.js*
- Using switch – See example: *lecture_codes/lesson3/using_switch.js*

- value is checked for strict equality to value of first case then second …
  - if equal equality execute code from corresponding case, until nearest break (or until the end of switch).
  - If no case matched then default code is executed (if it exists).

Exercise

- Write a program that asks user to enter number between 1 to 5 and prints out how the number is spelled.
  - First, write using else if
  - Then, refactor it to use switch

Nested If Statements

- There are times when we need to check for more conditions when prior conditions are met.
- One way to achieve this is using nested if statements.

```
let weather = prompt("Please enter weather outside");
let temp = prompt("Please enter current temperature");

if (weather == 'sunny') {
    if (temp < 80) {
        console.log("Good day for outdoor running")
    } else {
        console.log("Better use tread mill at home.")
    }
}
```

- Best to avoid more than one level of nesting if possible.

Scope of variables

- The scope of a variable determines how long and where a variable can be used.
- When const or let keywords are used, scope is within the block

```
let x = 5;
console.log(x);
if(x==5){
        let y = 2*x;
        console.log(y);
        console.log(x); // x is accessible here.
}
console.log(x);
console.log(y); // y is not accessible here.
```

- Declare y using `var` keyword in above code and see the change in output.

Exercise

- Write a program to compute sales commission based on following rules:
- If the salesman is salaried then
  - These is no commission for sales below $300
  - 1% for sales between $300 and $500 (exclusive)
  - 2% for sales above $500
- If the salesman is not salaried then
  - 2% for sales between $300 and $500 (exclusive)
  - 3% for sales above $500

Main points

- In programming, we encounter situations where we must make decisions based on some conditions, that determines the flow of code execution. We make use of relational and/or logical operators to make such decisions. Our programs produce expected results only when our decision logic is correct. *Science of Consciousness, as in programing, taking the right decision at the right point of time is also crucial to success in life. Our actions and decisions are spontaneously in the right direction when we have a good connection to the field of pure intelligence.*

Defining Table

An **algorithm** is simply a list of steps to perform some task. A large computer program contains many algorithms. Before creating an algorithm to solve a problem, you must be sure that you understand the problem. If you don't, you will probably create an algorithm that solves the wrong problem. A **defining table** is a useful tool to help you better understand a problem before you develop an algorithm to solve it. A defining table has three sections: input, processing, and output. To create a defining table, simply draw a table with the three sections. Then as you read and re-read the problem, put the parts of the problem into their

correct section in the table.

Example 1

You work for a large construction company. Your boss has asked you to write a computer program that will read a list of window openings for a building and compute, and output the total cost of all the windows. The window openings are entered in inches with the width first and the height second. The cost of a window is computed by multiplying the area of the window in square feet by $35.

Defining Table

Input	Processing	Output
A list of window openings For each window • width in inches • height in inches	For each window • compute area in sq. ft. • multiply area by $35 • add cost of this window to the total cost	total cost of all windows

References

- Conditional branching: if, '?' (javascript.info)
- The "switch" statement (javascript.info)