ANNOUNCEMENTS

- No posting of code in the forum
- Check class announcements daily
- JavaScript template (skeleton) available in “Resources” section of class web page
- Make sure you study this lecture’s material as we will have a lab next time
RESERVED WORDS (KEYWORDS)

- Reserved words → words you cannot use as identifiers (variable/function names)
- Some of them are:
  - var
  - break
  - do
  - if
  - catch
SPACES, SEMICOLONs, COMMENTS

- JavaScript ignores spaces, tabs, and newlines between tokens
- Use spaces to create nicely indented code
- The rules are usually one tab for indentation or three/four spaces. **You need to satisfy this requirement in programming assignments**
- A semicolon is generally used to mark the end of a statement and is optional when a statement appears on a separate line. For example, the following two set of statements are equivalent
  ```javascript
  x = 1;
y = 2;
x = 1
y = 2
  ```
- In this course we will always use a semicolon to mark the end of a statement
Comments in JavaScript

- Used to provide information to the programmer
- Used to identify sections in your code
- Ignored by the JavaScript interpreter

Two types of comments

- Inline comment ➔ // This is a comment until the end of the line
- Block comment
  /* The following is a comment that spans several lines */
  
  We can use a block comment for a single-line comment
  
  We can use comments to disable code
We can perform input and output via dialog boxes

- Input via `prompt`.
- **Example:** `InputOutput.html`
  - Notice we can define several variables at the same time
  - `prompt` is a function that displays a dialog box with the specified title. It can be used to read any data
  - You can read numbers and strings via `prompt`

- `prompt` → **returns a string**
  - If you need to perform some mathematical computation you might need to explicitly convert the value read it into a number
CONVERSIONS

- In JavaScript you don’t specify the type of variables
- Most of the time implicit transformations will take care of transforming a value to the expected one
- Example:

  ```javascript
  var x = "10";
  var y = "4";
  var a;
  a = x * y;  // Multiplying strings??
  document.writeln(a);
  ```

- Sometimes you might need to explicitly transform a value
- **Converting string to number**
  ```javascript
  var number = Number(stringValue);
  ```
- **Example:** Conversions1.html, Conversions2.html
Math.abs() → Absolute value
y = -10;
x = Math.abs(y);

Math.max() → Maximum of two values
x = 10;
y = 20;
a = Math.max(x, y);

Math.min() → Similar to maximum

Math.sqrt() → Square root
x = 4;
y = Math.sqrt(x);

Math.random() → Random value between 0 and less than 1
x = Math.random();

Constants
x = Math.PI; /* Mathematical constant pi */
BOOLEAN TYPE

- We have seen integer, float, and string values
- New type $\rightarrow$ boolean type
- Assumes the value *true* or *false*
- Variable declaration and initialization
  - `var found = true;`
  - `var attending = false;`
You can compare values by using the following operators

- `===`
  - Returns true if the values are equal, false otherwise (e.g., `x === y`)
- `!==`
  - Returns true if the values are different, false otherwise (e.g., `x !== y`)

Relational Operators

- `<` → Less than (e.g., `x < y`)
- `>` → Greater than (e.g., `x > y`)
- `<=` → Less than or equal (e.g., `x <= y`)
- `>=` → Greater than or equal (e.g., `x >= y`)

**Example:** Comparison1.html, Comparison2.html
If statement → Control statement that allow us to make decisions

First Form

```javascript
if (expression) {
    statement // executed if expression is true
}
```

Example: IfStm1.html

Second Form

```javascript
if (expression) {
    statement1 // executed if expression is true
} else {
    statement2 // executed if expression is false
}
```

Example: IfStm2.html