Chapter 17
Fundamental Concepts Expressed in JavaScript
What We’re Aiming for

figuring the price of espresso drinks
so baristas can have time to chat
Programming Concepts

• Names, values, and variables
• Declarations
• Data types, numbers, string literals, and Booleans
• Assignment
• Expressions
• Conditionals
Espresso Program

- Computes price of 4 kinds of espresso drinks based on:
  - Type of drink
  - Size of drink
  - Number of additional shots
  - Plus tax

Input:
- drink—a character string with one of the values: "espresso", "latte", "cappuccino", "Americano"
- ounce—an integer, giving the size of the drink in ounces
- shots—an integer, giving the number of shots

Output:
- price in dollars of an order, including 8.8% sales tax

Program:
1. var price;
2. var taxRate = 0.088;
3. if (drink == "espresso")
   price = 1.40;
4. if (drink == "latte" || drink == "cappuccino") {
   4a. if (ounce == 8)
      price = 1.95;
   4b. if (ounce == 12)
      price = 2.35;
   4c. if (ounce == 16)
      price = 2.75;
   }
5. if (drink == "Americano")
   price = 1.20 + .30 * (ounce/8);
6. price = price + (shots - 1) * .50;
7. price = price + price * taxRate;
Names, Values, And Variables

- Values change in programs

<table>
<thead>
<tr>
<th>Name</th>
<th>Current Value (7/1/2011)</th>
<th>Previous Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>U.S. President</td>
<td>Barack Obama</td>
<td>Bill Clinton, George H. W. Bush</td>
</tr>
<tr>
<td>Chief Justice</td>
<td>John Roberts</td>
<td>Warren Burger, Earl Warren</td>
</tr>
<tr>
<td>U.S. Supreme Court</td>
<td></td>
<td></td>
</tr>
<tr>
<td>James Bond</td>
<td>Daniel Craig</td>
<td>Sean Connery, Roger Moore</td>
</tr>
<tr>
<td>Queen of England</td>
<td>Elizabeth II</td>
<td>Victoria I, Elizabeth I</td>
</tr>
<tr>
<td>U.N. Secretary General</td>
<td>Ban Ki-moon</td>
<td>Boutros Boutros-Ghali, Kofi Annan</td>
</tr>
</tbody>
</table>
Variables are Names in a Program

- **Identifier** (variable) rules
  - Naming rules
    - Begin with a letter
    - May be followed by letter(s), digits (0-9), __
      (Cannot contain spaces)

<table>
<thead>
<tr>
<th>Valid</th>
<th>Invalid</th>
</tr>
</thead>
<tbody>
<tr>
<td>firstOne</td>
<td>1stOne</td>
</tr>
<tr>
<td>first1</td>
<td>first-1</td>
</tr>
<tr>
<td>First_1</td>
<td>first$1</td>
</tr>
<tr>
<td>First_One</td>
<td>first One</td>
</tr>
<tr>
<td>fIRSToNE</td>
<td>First1!</td>
</tr>
<tr>
<td>_special</td>
<td>5</td>
</tr>
<tr>
<td>very_long_name_ok</td>
<td>happy:)</td>
</tr>
</tbody>
</table>

For each *invalid* identifier, what rule is broken?
Variable Declaration

• What variables will be used in the program
  ```
  var radius, area;
  ```

• Used to calculate area of circle

• Type of statement

• Every variable used must first be declared

• Undefined value
Initialization

• Set initial value:

```javascript
var taxRate = 0.088;
```

• Related variables may be listed together

```javascript
var grade = 0, gpa = 0;
```

• Unrelated variables declared separately
3 Basic Data Types: Numbers, Booleans, Strings

• Numbers
  – No "units"
  – About 10 significant digits (10^{-324} to 10^{308})

• Booleans
  – Values: true and false
  – May initialize variables

• Strings
  – Sequences of keyboard characters
  – Surrounded by single ‘ or double “ quotes

  ```javascript
  var greet = 'I asked “How are you?”'
  ```
Rules for Writing Strings

- Surrounded by single ‘ or double “ quotes
- Allow most characters except return ( ), backspace, tab, \
- Double quoted strings can contain single quoted strings and vice versa. `var greet = “ She said ‘Hi!’ ”;`
- Any number of characters allowed
- Minimum number of characters is zero "" (empty string)

<table>
<thead>
<tr>
<th>Sequence</th>
<th>Character</th>
<th>Sequence</th>
<th>Character</th>
</tr>
</thead>
<tbody>
<tr>
<td>\b</td>
<td>Backspace</td>
<td>\f</td>
<td>Form feed</td>
</tr>
<tr>
<td>\n</td>
<td>New line</td>
<td>\r</td>
<td>Carriage return</td>
</tr>
<tr>
<td>\t</td>
<td>Tab</td>
<td>'</td>
<td>Apostrophe or single quote</td>
</tr>
<tr>
<td>&quot;</td>
<td>Double quote</td>
<td>\</td>
<td>Backslash</td>
</tr>
</tbody>
</table>
Assignment Statement

- Changes variable’s value

variable: `weeks`  
expression: `days / 7`  
operator: `/`

Step 1) Evaluate right side
Step 2) Assign to left side

```
weeks = days / 7;
```
Arithmetic Operators

• Binary operators:  +  -  *  /  %
  – Modulus ( % ) returns the remainder
    e.g.  6 % 2  is  0
         7 % 2  is  1

• Unary operators:  -  ++  --
  e.g.  -3

• Increment: increase integer value by one
  e.g.  quiz = 3;
       quiz++;  will give 4

• Decrement: decrease integer value by one
  e.g.  quiz = 3;
       quiz--;  will give 2
## Relational Operators

<table>
<thead>
<tr>
<th>Operation</th>
<th>Result</th>
</tr>
</thead>
<tbody>
<tr>
<td>3 &lt; 4</td>
<td>true (1)</td>
</tr>
<tr>
<td>3 &lt;= 4</td>
<td>true (1)</td>
</tr>
<tr>
<td>3 == 4</td>
<td>false (0)</td>
</tr>
<tr>
<td>3 != 4</td>
<td>true (1)</td>
</tr>
<tr>
<td>3 &gt; 4</td>
<td>false (0)</td>
</tr>
<tr>
<td>4 &gt;= 4</td>
<td>true (1)</td>
</tr>
<tr>
<td>'A' &lt; 'B'</td>
<td>true (1)</td>
</tr>
</tbody>
</table>

Characters are evaluated in alphabetical order.
Logical Operators

<table>
<thead>
<tr>
<th>Logical operator</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>&amp;&amp;</td>
<td>And</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td>!</td>
<td>Not</td>
</tr>
</tbody>
</table>

• Test two or more relationships
  – Teenagers are older than 12 and younger than 20

| x     | ! x   | x || y | X && y |
|-------|-------|--------|--------|
| true  | false | true   | true   |
| true  | false | true   | false  |
| false | true  | true   | false  |
| false | false | false  | false  |
if Statement

- General form:

```java
if ( <Boolean expression> )
statement(s)
```

```c
if ( ( num % 2 ) == 0 )
even = true ;
```
if-else Statement

• General form:

  if ( <Boolean expression> )
  statement(s)
else
  statement(s)

```java
if( ( num % 2 ) == 0 )
  even = true;
else
  even = false;
```
if (flip1 == guess1) {
    if (flip2 == guess2)
        score = "win win";
    else
        score = "win lose";
} else {
    if (flip2 == guess2)
        score = "lose win";
    else
        score = "lose lose";
}
The Espresso Program: Code

```javascript
var drink = "latte";
var ounce = 12;
var shots = 2;

var price;
var taxRate = 0.088;

if (drink == "espresso")
    price = 1.40;
if (drink == "latte" || drink == "cappuccino")
{
    if (ounce == 8)
        price = 1.95;
    if (ounce == 12)
        price = 2.35;
    if (ounce == 16)
        price = 2.75;
}

if (drink == "Americano")
    price = 1.20 + 0.30*(ounce/8);
price = price + (shots - 1)*.50;
price = price + price*taxRate;
```
1. var price;
2. var taxRate = 0.088;
3. if (drink == "espresso")
   price = 1.40;
4. if (drink == "latte" || drink == "cappuccino") {
4a. if (ounce == 8)
     price = 1.95;
4b. if (ounce == 12)
     price = 2.35;
4c. if (ounce == 16)
     price = 2.75;
}
5. if (drink == "Americano")
   price = 1.20 + .30 * (ounce/8);
6. price = price + (shots - 1) * .50;
7. price = price + price * taxRate;
<input type="button" value="Total" onclick="var price;
var taxRate = 0.088;
if (drink == "espresso")
  price = 1.40;
if (drink == "latte" || drink == "cappuccino") {
  if (ounce == 8)
    price = 1.95;
  if (ounce == 12)
    price = 2.35;
  if (ounce == 16)
    price = 2.75;
}
if (drink == "Americano")
  price = 1.20 + .30 * (ounce/8);
price = price + (shots - 1) * .50;
price = price + price * taxRate;
document.forms[0].price.value = price;
//one more assignment is required here
"/>
the bean counter

figuring the price of espresso drinks so baristas can have time to chat
the bean counter

figuring the price of espresso drinks so baristas can have time to chat

1 S ESPRESSO
2 T LATTE
3 G CAPPUCCINO
4 AMERICANO 0.00
Arithmetic Operators

• (*) Multiply operator

• Multiply and divide are performed before add and subtract
  – Unless grouped by parentheses \((2 + 3) * 4\)

• JavaScript has no operator for exponents \((3 * 3)\)

• Binary operators operate on two values \((2 + 3)\)

• Unary operators operate on one value \((-3)\)

• Modulus or mod \((\%\)) divides, returns remainder
Overloaded Operators

• Operator Overload
  – Use of an operator with different data types
  – Adds with numbers
    \[ 4 + 5 \text{ produces } 9 \]
• Concatenation with strings
  \[ "four" + "five" \rightarrow "fourfive" \]
<!doctype html>
<html>
<head>
<meta charset="UTF-8">
<title>Bean Counter</title>
</head>
<body>
<h2>Confirming that bean.html works</h2>
<script>
var drink = "latte";
var ounce = 12;
var shots = 2;
var taxRate = 0.088;
var price;
if (drink == "espresso")
    price = 1.40;
if (drink == "latte" || drink == "cappuccino") {
    if (ounce == 8)
        price = 1.95;
    if (ounce == 12)
        price = 2.35;
    if (ounce == 16)
        price = 2.75;
}
if (drink == "Americano")
    price = 1.20 + 0.30*(ounce/8);
price = price + (shots - 1)*.50;
price = price + price*taxRate;
alert(price);
</script>
</body>
</html>