TA Lecture Notes

September 15, 2014

Announcements

- Having JavaScript problems? Check Piazza!
- Please don't post code on Piazza
- Weekly activity 4 is out today, due Monday, September 22 by 9am
- i>clickers should be registered on the CS105 website

Previously, we had introduced a code example that reads in prices of items and returns their total:

```
var total = 0;
var price = readline();       // readline() returns a string

total = total + (price * 1); // multiplying by 1 converts the
                             // variable price into a number

while (price != 0)           // loop: no semicolons
{
      price = readline();
      total = total + (price * 1);
}

print("The total of your items is: " + total);
```

`readline()` and `print()` are **functions** that are defined somewhere. The computer already knows what to do with them.

- We can also define our own functions!
- They can be reused as many times as we want

Function definition example

```
function readNumber() // functions are declared as name(parameters)
{
      var s = readline();
      return (s * 1);  // return statements are used if you want the
                       // function to give back a value
}
```

In this case, the function gives a number version of what was read by using `readline()`.

How do we edit our previous code to use this new function?

```
var total = 0;
var price = readNumber();

total = total + price;

while (price != 0)
{
      price = readNumber();
      total = total + price;
}

print("The total of your items is: " + total);
```

What about bad inputs?

- Typing "five" instead of "5" results in `NaN`, which means ***not a number***
- We can fix things so that if the input is not a number, the code can still handle it
- Generally, we should avoid using `(s * 1)`, as it doesn't actually check whether or not `s` is a number

Numbers in JavaScript

- **Integers ("ints")**: any whole number; either positive, negative, or zero
- **Floating point numbers ("floats")**: any non-integer, real number – represented approximately as best as the computer can
- In JavaScript, there is a built-in function called **`parseFloat(s)`**
  - You can look up the details of built-in JavaScript functions online using Google
  - In our function, simply return `parseFloat(s);`
  - How do you check if it's a number?: use the function **`isNaN(testValue);`** where `testValue` is the value you're testing

---

`readNumber()` example using `parseFloat(s)` and `isNaN(testValue)`

```
function readNumber()
{
      var s = readline();
      var f = parseFloat(s);
      while (isNaN(f) == true) // use while, not if, to keep asking
                               // while the inputs are constantly bad
      {
            print("Try again");
            s = readline();
            f = parseFloat(s);
      }
      return f;
}
```

Example: making change

```
print("How many cents do you want back?");
var cents = readNumber();
var centsLeft = cents;

var q = 0;                // quarters
while (centsLeft >= 25) // give maximum number of quarters that you can
{
      q = q + 1;
      centsLeft = centsLeft - 25;
}

var d = 0;                 // dimes
while (centsLeft >= 10)  // give maximum number of dimes that you can
                           // from the remaining change
{
      d = d + 1;
      centsLeft = centsLeft - 10;
}

var n = 0;              // nickels
while (centsLeft >= 5) // give maximum number of nickels that you can
                       // from the remaining change
{
      n = n + 1;
      centsLeft = centsLeft - 5;
}

var p = centsLeft;     // pennies

print("Q: " + q);
print("D: " + d);
print("N: " + n);
print("P: " + p);
```