

Section Handout #8: Objects

Portions of this handout by Jason Chen, Kat Gregory, and Eric Roberts

You've made it! Congratulations! This is our last section, and once again I would like to thank everyone for all the hard work you've put in all quarter. The teaching staff have had an amazing time learning with all of you, and we wish you the very best!

Problem 1: Discussion of Different Flavors of JavaScript Objects

As I mentioned during my first lecture on objects, JavaScript overloads the word "object" with a number of different meanings. We partitioned these different "flavors" of objects into three lectures: objects as aggregates, objects as maps, and objects in the Object Oriented Programming (OOP) paradigm. Let's review them.

For each type of object, brainstorm the defining characteristics that distinguish it. In what kind of situations is it useful? What makes it different from the other object types? What are some classic examples of this object type?

Objects as Aggregates:

Objects as Maps:

Objects in OOP:

Problem 2: Roman Numerals

In Roman numerals, characters of the alphabet are used to represent integers as shown in this table:

symbol	value
I	1
V	5
X	10
L	50
C	100
D	500
M	1000

Each character in a Roman numeral stands for the corresponding value. Ordinarily, the value of the Roman numeral as a whole is the sum of the individual character values given in the table. Thus, the string "LXXVI" denotes $50 + 10 + 10 + 5 + 1$, or 76. The only exception occurs when a character corresponding to a smaller value precedes a character representing a larger one, in which case the value of the first letter is subtracted from the total, so that the string "IX" corresponds to $10 - 1$, or 9.

Write a function `romanToDecimal` that takes a string representing a Roman numeral and returns the corresponding decimal number. To find the values of each Roman numeral character, your function should look that character up in a map that implements the Roman numeral conversion. If the string contains characters that are not in the table, `romanToDecimal` should return -1 .

Problem 3: Morse Code

In May of 1844, Samuel F. B. Morse sent the message "What hath God wrought!" by telegraph from Washington to Baltimore, heralding the beginning of the age of electronic communication. To make it possible to communicate information using only the presence or absence of a single tone, Morse designed a coding system in which letters and other symbols are represented as coded sequences of short and long tones, traditionally called *dots* and *dashes*. In Morse code, the 26 letters of the alphabet are represented by the codes shown below.

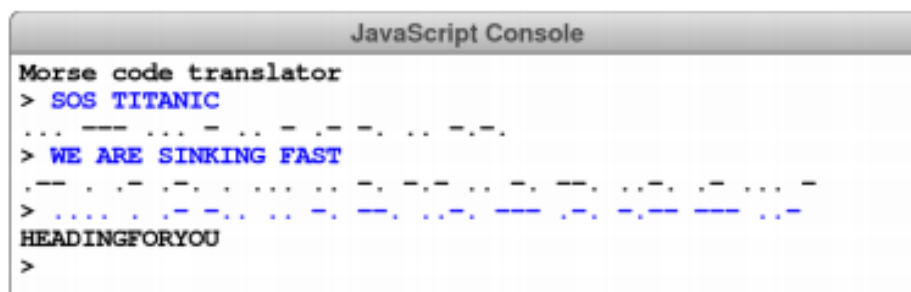
FIGURE 9-14 Morse code

A	• —	H	• • • •	O	— — —	V	• • • —
B	— • • •	I	• •	P	• — — •	W	• — —
C	— • — •	J	• — — —	Q	— — • •	X	— • • •
D	— • •	K	— • —	R	• — •	Y	— • — —
E	•	L	• — • •	S	• • •	Z	— — • •
F	• • — •	M	— —	T	—		
G	— — •	N	— •	U	• • —		

Write a program that reads in lines from the user and translates each line either to or from Morse code, depending on the first character of the line:

- If the line starts with a letter, you want to translate it to Morse code. Any characters other than the 26 letters should simply be ignored.
- If the line starts with a period (dot) or a hyphen (dash), it should be read as a series of Morse code characters that you need to translate back to letters. You may assume that each sequence of dots and dashes in the input string will be separated by spaces, and you are free to ignore any other characters that appear. Because there is no encoding for the space between words, the characters of the translated message will be run together when your program translates in this direction.

The program should end when the user enters a blank line. A sample run of this program (taken from the messages between the Titanic and the Carpathia in 1912) might look like this:



```
JavaScript Console
Morse code translator
> SOS TITANIC
... --- ... --- ... --- ... --- ... ---
> WE ARE SINKING FAST
... --- ... --- ... --- ... --- ... ---
> HEADINGFORYOU
... --- ... --- ... --- ... --- ... ---
>
```

Problem 4: Review

Come to section with any questions you have from the material we covered the last few weeks! As a reminder, the final exam Monday, December 10th from 8:30 – 11:30am in 380-380C. We will be holding a two-hour review session the morning of Saturday, December 8th at 11:00am in McCullough 115.

Lastly, if you are interested in where you can go from here on out, please come talk to any of us. We would love to point you in the right direction based on your interests. There's so much out there and CS106AJ was just the beginning!