Intekglobal

JavaScript

Exercise (Chapter 3~14)

Time Taken: 76 hours

Chapter 03

(4 hours)

1. Create a function that will multiply two numbers. The function must return the result in base 13.

   Example:

   var answer = mul (9, 6); // 42

   Basic function creation together with built-in JS Math functionality.

   Estimated Time: 1 hours.

2. Create a function that will return the addition of N numbers.

   Example:

   ```
   var answer = add (1, 2) + add ( 1, 4, 6, 7, 2);
   ```

   Dynamic argument list handling with JS

   Estimated Time: 10 min<.

3. Create an object that will hold methods for adding, multiplying, and factorial operations.

   Example:

   ```
   var a = myMath.add (1, 2, 3); //6
   var b = myMath.mul (1, 2, 3); // 6
   var c = myMath.fact (  3); // 6
   ```

   Use of anonymous functions and functions as object properties (methods.)

   Estimated Time: 15 min<.

4. Create a custom object that will hold an image's mock information such as pixel color data, image size, and name. It must be able to return the information.

   Example:

   ```
   var data = new Array (1600); // 40 x 40 px dummy image data
   var img = new Image (data, 40, 40, 'myImage');
   img.width; // 40
   img.height; // 40
   img.name; // 'myImage'
   img.getPixel(20, 4); // returns the color of the pixel at that position.
   ```

   Estimated Time: 1 hour.

5. Create a function that will print out the properties of an object.
   a. If one parameter is provided, it should print out all of the properties accessible by that object.
   b. If a second boolean value parameter is provided, it should only print out the values that belong to the object instance itself if true.

   Example:

   ```
   function CustomObject (a, b) {
       this.a = a;
       this.b = b;
   }
   ```

```
CustomObject.prototype.c = function () { return this.a + this.b; };
var obj = new CustomObject (1, 2);
printObjProp (obj); // output: a, b, c
printObjProp (obj, false); // output: a, b, c
printObjProp (obj, true); // output: a, b
```

Estimated Time: 1 hour<.

Chapter 04

(15 hours)

1. Create a recursive function that will calculate the fibonacci value of a number.
   Example:
   ```
   var n = fibonacci (4); // 3
   var m = fibonacci (9); // 34
   ```
   Use of recursion.
   Estimated Time: 1 hr <.

2. Create a function that will recursively go through all of the elements of an array of numbers and add them.
   Example:
   ```
   var arr = [ 1, 3, 5, 7];
   var sum = addRec (arr); // 16
   ```
   Use of recursion.
   Estimated Time: 2 hr.

3. Create a custom object type that will hold a number value.
   a. Make sure that no other data type can be assigned to that variable.
   b. It must hold ONLY numbers.

   Hint: Validation, setters & getters, private variables
   Estimated Time: 2 hr <.

4. Write a function that will accept any number of arguments and print out their data type.
   Example:
   ```
   dataType (1, 6.2831, "pi*2", [function(){}, 1], {}, function () {});
   // number, float, string, array, object, function
   ```
   Variable argument functions & data types
   Estimated Time: 2 hr.

5. Write a function that will calculate the distance between two points. The function should be able to handle 2D and 3D points.
   Example:
   var **x1** = 1, **y1** = 2, **z1** = 1;
   var **x2** = 2, **y2** = 2, **z2** = 4;
   var delta1 = distance (x1, y1, x2, y2); // delta = 1
   var delta2 = distance (x1, y1, z1, x2, y2, z2); // delta = 3.1622…

distance (x1, x2); // should throw an error: Insufficient parameters

Function overloading and validation

Estimated Time: 4 hr.

6. Make the function from exercise 5 accept its parameters as either a parameter list or as two arrays containing 2D or 3D point data.

   Example:

   distance (1, 2, 2, 2); // returns 1 (done as part of exercise 5)
   distance ([1,2], [2,2]); // returns 1
   distance ([1,2], [2,2,4]); // error: incompatible point data

   Function overloading and validation

   Estimated Time: 4 hr.

Chapter 05

(8 hours)

1. Implement the following:
   a. A bank that holds client's information:
      i. account number;
      ii. balance
   b. A set of clients where each can:
      i. hold money of their own;
      ii. deposit money into the bank (to any account);
      iii. retrieve money from the bank (from personal account only);
      iv. view current balance in bank (from personal account only)
   c. A client cannot deposit more money than what it has;
   d. A client cannot retrieve more money that what is in its account;
   e. All financial information must be private

   Use of closures

   Estimated Time: 4 hr.

2. Implement the following:
   a. A large building has many people and pieces of equipment. A new tech-support employee has been hired to help out solve users' problems and fix broken equipment. The new employee is still unfamiliar with the layout but is doing his best to keep track of where everyone and everything is.
      i. Implement a structure that represents the building.
         1. Must contain data types representing equipment and users.
      ii. Equipment can be associated with rooms or specific people
      iii. Each piece of equipment and person is associated with a specific floor and room.
      iv. The new tech-support employee must be able to find users and equipment as quickly as possible.

v. The new employee must remember past searches

Use of memoization and implementation of a simple searching algorithm.
Estimated Time: 4 hr.

Chapter 06
(4 hours)

1. Create a set of object types that describe a series of related objects that may share behavior and/or attributes. Code the example and another set of classes different from the example. Add properties/methods as needed.
   Example:
   Shape { pEdges, fnDisplay }
   Quadrilateral is Shape { fnArea, fnPerimeter}
   Square is Quadrilateral { }
   Triangle is Shape {fnArea, fnPerimeter}
   Use of inheritance, prototype, and function overwriting.
   Estimated Time: 4 h.

Chapter 07
(2 hours)

1. Create a function that will transform a hex number into an rgb format.
   Example:
   “#3020ff” → “rgb (48, 32, 255)”
   Use of regular expressions
   Estimated Time: 30 min<.

2. Create a function that will transform a U.S style date format into a format of a different language/region. If that date is a holiday in the target locale (language & region), it should be mentioned. *Preferably, use a different language from that of the example.*
   Example:

   English-US: 09/16/2014 → Spanish-MX: 16/09/2014 (Dia de la independencia)
   Spanish-MX: 1/4/2014 → English-US: 4/1/2014 (April fools day)
   Use of regular expressions, and lookup tables
   Estimated Time: 1 hr.

Chapter 08
(1 hour)

1. Create a function that will display a random sentence to the console every minute.
   Basic use of timers
   Estimated Time: 10 min <.

2. Create functions A, B, and C that execute every 30s, 1min, and 1min 15s respectively. Use only 1 timer/interval to control all three functions.
   use of a central timer controller
   Estimated Time: 30 min.

Chapter 09

(2 hours)

1. Create a function that receives a parameter containing a string. The string data must be transformed into an object that can hold properties and methods. *Note: Some changes may be required in the string data*
   Example:
   var str = "{prop1: 42, myFn: function(a, b) { return a+b+this.prop1;}}"

   var obj = dataParse(str);
   code evaluation
   Estimated Time: 2 hr.

03/26/2014 Line break

Chapter 10

(4 hours)

1. Given the object structure bellow, do the following:
   a. simplify the function calls by using the ***with*** statement; and
   b. simplify the function calls without using the ***with*** statement.

Object structure

```
var myLib = {
        math: {
                real: {
                        add: function (a, b){ /*code goes here*/},
                        sub: function (a, b){ /*code goes here*/},
                        mul: function (a, b){ /*code goes here*/}
                },
                complex: {
                        Num: function (real, img){/*code goes here*/},
                        add: function (a, b){ /*code goes here*/},
                        sub: function (a, b){ /*code goes here*/},
                        mul: function (a, b){ /*code goes here*/}
               },
               matrix: {
                        add: function (a, b){ /*matrix addition*/},
                        sub: function (a, b){ /*matrix subtraction*/},
                        mul: function (a, b){ /*matrix multiplication*/},
                        eye: function (size){ /*identity matrix*/},
                        dot: function (m, a){ /*dot product*/},
                        times:function(a, b){ /*element-wise multiplication*/}
                }
        }
};
```

Function calls

```
var answer = myLib.math.real.sub(
                  myLib.math.real.add (20, 22),
                  myLib.math.real.mul(2,5));

var ans = myLib.math.matrix.times(
              myLib.math.matrix.eye (4),
              myLib.math.complex.sub (
                      new myLib.math.complex.Num (
                             myLib.math.real.add(5,2),
                             -3),
                      new myLib.math.complex.Num (3,4)
              )
          );
```

Use of the with statement and alternatives to it.
Estimated Time: 4 hr.

Chapter 11

N/A (Cross browser issues)

Chapter 12

(4 hours)

1. Given the following HTML markup, create a JS function that can print out the attributes of the DOM elements:

   <div id='a' class='square' style='display:inline-block' val='something important'></div>

   Example:

   printAttr (el, ['id', 'class', 'style', 'val']);
   // should print out:
   // a
   // square
   // display:inline-block
   // something important
   Use of built in JS functionality for attribute retrieval
   Estimated Time: 4 hr.

Chapter 13

(8 hours)

1. Create a 5 x 5 grid (use tables or divs.) Each element of the grid will be sequentially numbered and each element must respond to a click event by alerting its corresponding number.

Example: 2x2 grid

0	1
2	3

Event assignment, and delegation.
Estimated Time: 4 hr.

2. Create a button that can only be used 3 times. Indicate the usage of the button using the meter element. Once the button has been pressed three times, it must not activate again.
   Removing events, closures, and form element usage.
   Estimated Time: 4 hr.

Chapter 14

(16 hours)

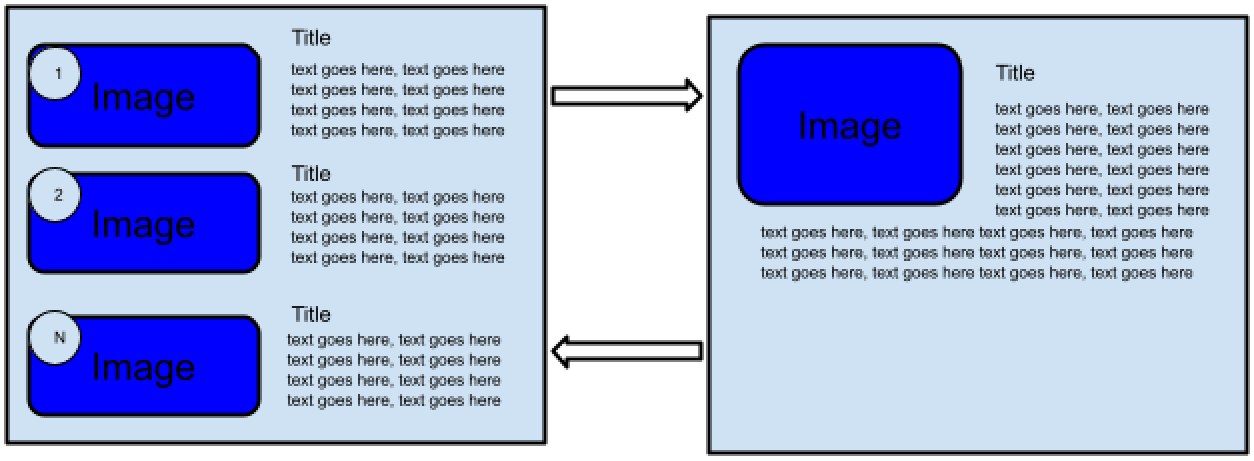
1. Create a function that will populate a containing div with a grid of NxM size. The size must be user defined through a GUI. Every time a new size is selected, any information already on the container should be reset.
   Dom manipulation through user input
   Estimated Time: 4 hr.

2. Given a reasonably sized piece of text, have it displayed on the body of a page in 2, 3, or 4 columns. A user must be able to change the number of columns the text is displayed on.
   Dom manipulation, CSS formatting
   Estimated Time: 4 hr.

3. Create a simple Single Page Application (SPA) that will have two types of templates. Specifications:
   a. Template 1: should display an image on the left with a title and snippet of text on the right.
   b. Template 2: should display the image on the top left, and the title and text should wrap around the image.
   c. Templates should be created on custom script tags or template tags
   d. fragments should be used where applicable
   e. The landing page should be template 1
   f. From template 1, clicking on an image or the text associated with that image should result in navigating to template 2.
   g. From template 2, clicking on a link should return the view to template 1
   h. Items from template 1 should be numbered using CSS rules
   i. The data that will populate the templates should be read from a JSON file
   j. The current view (template 1 or template 2) should be maintained even if the browser is refreshed.

k. The user should be able to directly navigate to any of the page's views

Example:



Use of   fragments, and dom manipulation
Estimated Time: 8 hr.