EXAMPLE 40: NESTEP FUNCTIONS

IT HAPPENS ALL THE TIME.

A FEW LINES OF COPE INSIDE A FUNCTION THAT YOU COPY-PASTE OVER.

EVERYONE POES IT - BUT ITS BAP PROGRAMMING PRACTICE.

EVERYONE POES IT - BUT ITS BAP PROGRAMMING PRACTICE.

NESTED FUNCTIONS CAN MAKE A REAL DIFFERENCE HERE.

THIS IS ESPECIALLY TRUE IN JAVASCRIPT, WHERE CODE READABILITY IS A REAL PROBLEM.

```
function Rectangle(l,b) {
 this.length = 3; ED FUNCTIONS this.breadth So; ED FUNCTIONS
window.onload = function(){
  var rectangle1 = new Rectangle(3,4);
  var rectangle2 = new Rectangle(4,5);
  var rectangle3 = new Rectangle(5,6);
  var rectArray = [rectangle1, rectangle2, rectangle3];
  //printStuffAboutRectangleArray(rectArray);
  var printStuffAboutRectangleArray = function(rectangleArray) {
    var getArea = function(rectangle){
      console.log("Inside a nested function that calculates the area of a
rectangle");
      return rectangle.length * rectangle.breadth;
    for (var i = 0;i<rectangleArray.length;i++) {</pre>
       var r = rectangleArray[i];
       console.log(r.length + "," + r.breadth + "," + getArea(r));
  }
  printStuffAboutRectangleArray(rectArray);
  // the nested function getArea will not exist here!
```

```
function Rectangle(l,b) {
 this.length = 3; ED FUNCTIONS this.breadth So; ED FUNCTIONS
window.onload = function(){
  var rectangle1 = new Rectangle(3,4);
  var rectangle2 = new Rectangle(4,5);
  var rectangle3 = new Rectangle(5,6);
  var rectArray = [rectangle1, rectangle2, rectangle3];
  //printStuffAboutRectangleArray(rectArray);
  var printStuffAboutRectangleArray = function(rectangleArray) {
    var getArea = function(rectangle){
      console.log("Inside a nested function that calculates the area of a
rectangle");
      return rectangle.length * rectangle.breadth;
    for (var i = 0;i<rectangleArray.length;i++) {</pre>
       var r = rectangleArray[i];
       console.log(r.length + "," + r.breadth + "," + getArea(r));
  }
  printStuffAboutRectangleArray(rectArray);
  // the nested function getArea will not exist here!
```

```
function Rectangle(l,b)
                        WE HAVE A SIMPLE OBJECT
  this.length = l;
  this.breadth = b;
                                 CONSTRUCTOR
window.onload = funttion(){
  var rectangle1 = new Rectangle(3,4);
  var rectangle2 = new Rectangle(4,5);
  var rectangle3 = new Rectangle(5,6);
  var rectArray = [rectangle1, rectangle2, rectangle3];
  //printStuffAboutRectangleArray(rectArray);
  var printStuffAboutRectangleArray = function(rectangleArray) {
    var getArea = function(rectangle){
      console.log("Inside a nested function that calculates the area of a
rectangle");
      return rectangle.length * rectangle.breadth;
    for (var i = 0;i<rectangleArray.length;i++) {</pre>
       var r = rectangleArray[i];
       console.log(r.length + "," + r.breadth + "," + getArea(r));
  printStuffAboutRectangleArray(rectArray);
  // the nested function getArea will not exist here!
```

```
function Rectangle(l,b) {
  this.length = l;
  this.breadth = b;
window.onload = funcWE(CREATE A FEW DIFFERENT OBJECTS
  var rectangle1 = new rectangle2 = new Rectangle (4,5)$ OBJECT CONSTRUCTOR
  var rectangle3 = new Rectangle(5,6);
  var rectArray = [rectangle1], rectangle2, rectangle3];
  //printStuffAboutRectangleArray(rectArray);
var NOW, LET'S SAY-THEREIS SOME SIMPLE OPERATIONay) {
          WE'D LIKE TO PERFORM WITH ALL OF THESE.
      console.log("Inside a nested function that calculates the area of a
rectangle");
      return rectangle.length * rectangle.breadth;
    for (var i = 0;i<rectangleArray.length;i++) {</pre>
       var r = rectangleArray[i];
       console.log(r.length + "," + r.breadth + "," + getArea(r));
  printStuffAboutRectangleArray(rectArray);
  // the nested function getArea will not exist here!
```

WE HAVE 3 OPTIONS

- ADD AN OBJECT PROPERTY (TO THE OBJECT) FOR THIS SIMPLE
- SET UP A NESTED FUNCTION, AND APPLY IT TO EACH OF THE OBJECTS
- JUST TYPE OUT THE CODE FOR THIS SIMPLE OPERATION 3 TIMES, ONCE PER OBJECT WE CREATED

WE HAVE 3 OPTIONS

- ADD AN OBJECT PROPERTY (TO THE OBJECT) FOR THIS SIMPLE ACTION OVERKILL INVOLVES CHANGING THE OBJECT
- SET UP A NESTED FUNCTION, AND APPLY IT TO EACH OF THE OBJECTS
- JUST TYPE OUT THE COPE FOR THIS SIMPLE OPERATION 3 TIMES, ONCE PER OBJECT WE CREATED

WE HAVE 3 OPTIONS

- APP AN OBJECT PROPERTY (TO THE OBJECT) FOR THIS SIMPLE ACTION
- SET UP A NESTED FUNCTION, AND APPLY IT TO EACH OF THE OBJECTS
- JUST TYPE OUT THE CODE FOR THIS SIMPLE OPERATION 3 TIMES, ONCE PER OBJECT WE CREATED

VERY HACKY - OVER TIME, COPY-PASTING CODE LEADS TO PROBLEMS.

WE HAVE 3 OPTIONS

- ADD AN OBJECT PROPERTY (TO THE OBJECT) FOR THIS SIMPLE ACTION
- SET UP A NESTED FUNCTION, AND APPLY IT TO EACH OF THE OBJECTS

AHA! PERFECT, THE BEST MIDDLE OF THE ROAD SOLUTION!

 JUST TYPE OUT THE COPE FOR THIS SIMPLE OPERATION 3 TIMES, ONCE PER OBJECT WE CREATED

```
function Rectangle(l,b) {
 SET UP A NESTED FUNCTION, AND APPLY IT TO EACH OF THE OBJECTS
window.onload = function(){
 var rectangle1 = new Rectangle(3,4);
var rectangle2 = nACTUALLY,eWEDO EVEN BETTER - WE BUILD A
 var rectangle3 = NESTED FUNCTION INSIDE A NESTED FUNCTION!!
  var rectArray = [rectangle1, rectangle2, rectangle3];
  //printStuffAboutRectangleArray(rectArray);
  var printStuffAboutRectangleArray = function(rectangleArray) {
    var getArea = function(rectangle){
      console.log("Inside a nested function that calculates the area of a
rectangle");
      return rectangle.length * rectangle.breadth;
    for (var i = 0;i<rectangleArray.length;i++) {</pre>
       var r = rectangleArray[i];
       console.log(r.length + "," + r.breadth + "," + getArea(r));
  }
  printStuffAboutRectangleArray(rectArray);
  // the nested function getArea will not exist here!
```

```
this.lengtACTUALLY, WE DO EVEN BETTER - WE BUILD A
  this break ESTED FUNCTION INSIDE A NESTED FUNCTION!!
window.onload = function(){
  var rectangle1 = new Rectangle(3,4);
  var rectangle2 = new Rectangle(4,5);
 var rectangle3 = new OUTER NESTED FUNCTION, APPLY IT
 var rectArray = [rectangle1_rectangle2_rectangle2_rectangle3]r OBJECTS
//printStuffAboutRectangleArro (ANAAKKAY OF OBJECTS
 var printStuffAboutRectangleArray = function(rectangleArray)
    var getArea = function(rectangle){
      console.log("Inside a nested function that calculates the area of a
rectangle");
      return rectangle.length * rectangle.breadth;
    for (var i = 0;i<rectangleArray.length;i++) {</pre>
       var r = rectangleArray[i];
       console.log(r.length + "," + r.breadth + "," + getArea(r));
  printStuffAboutRectangleArray(rectArray);
  // the nested function getArea will not exist here!
```

```
this.lengtACTUALLY, WE DO EVEN BETTER - WE BUILD A
  this breakESTED FUNCTION INSIDE A NESTED FUNCTION!!
window.onload = function(){
  var rectangle1 = new Rectangle(3,4);
  var rectangle2 = new Rectangle(4,5);
 var rectangle3 = new OUTER NESTED FUNCTION, APPLY IT
 var rectArray = [rectangle1_rectangle2_rectangle2_rectangle3] F OBJECTS //printStuffAboutRectangleArro (ANAAKKAY OF OBJECTS
 var printStuffAboutRectangleArray = function(rectangleArray)
    var getArea = function(rectangle){
      console.log("Inside a nested function that calculates the area of a
rectangle");
      return rectangle length * rectangle breadth; INNER NESTED FUNCTION, APPLY IT
       var r = TO EACH OBJECT OF THE ARRAY!
  printStuffAboutRectangleArray(rectArray);
  // the nested function getArea will not exist here!
```

```
this.lengtACTUALLY, WE DO EVEN BETTER - WE BUILD A
  this breakested function inside A nested function!!
window.onload = function(){
  var rectangle1 = new Rectangle(3,4);
  var rectangle2 = new Rectangle(4,5);
 var rectangle3 = new OUTER NESTED FUNCTION, APPLY IT
 var rectArray = [rectangle1_rectangle2_rectangle2_rectangle3]r OBJECTS
//printStuffAboutRectangleArro (ANAAKKAY OF OBJECTS
 var printStuffAboutRectangleArray = function(rectangleArray)
    var getArea = function(rectangle){
   INNER NESTED FUNCTION, APPLY IT TO EACH OBJECT tes the area of a
             rectangle:length * rectangle:breadth;
    for (var i = 0;i<rectangleArray.length;i++) {</pre>
       var r = rectangleArray[i];
       console.log(r.length + "," + r.breadth +
  printStuffAboutRectangleArray(rectArray);
  // the nested function getArea will not exist here!
```

```
this.lengtACTUALLY, WE DO EVEN BETTER - WE BUILD A
  this break ESTED FUNCTION INSIDE A NESTED FUNCTION!!
window.onload = function(){
  var rectangle1 = new Rectangle(3,4);
  var rectangle2 = new Rectangle(4,5);
 var rectangle3 = new OUTER NESTED FUNCTION, APPLY IT
 var rectArray = [rectangle1_rectangle2_rectangle2_rectangle3] F OBJECTS //printStuffAboutRectangleArro (ANAAKKAY OF OBJECTS
  var printStuffAboutRectangleArray = function(rectangleArray)
    var getArea = function(rectangle){
   INNER NESTED FUNCTION. APPLY IT TO EACH OBJECT
      <del>return rectangle:length * rec</del>tangle.breadth;
    for (var i = 0;i<rectangleArray.length;i++) {</pre>
       var r = rectangleArray[i];
       console.log(r.length + "," + r.breadth + "," + getArea(r));
  printStuffAboutRectangleArray(rectArray);
  // the nested function getArea will not exist here!
```

```
this.lengtACTUALLY, WE DO EVEN BETTER - WE BUILD A
  this break ESTED FUNCTION INSIDE A NESTED FUNCTION!!
window.onload = function(){
  var rectangle1 = new Rectangle(3,4);
  var rectangle2 = new Rectangle(4,5);
 var rectangle3 = new OUTER NESTED FUNCTION, APPLY IT
 var rectArray = [rectangle1_rectangle2_rectangle2_rectangle3]r OBJECTS
//printStuffAboutRectangleArro (ANAAKKAY OF OBJECTS
 var printStuffAboutRectangleArray = function(rectangleArray)
    var getArea = function(rectangle){
                                                          tes the area of a
   INNER NESTED FUNCTION, APPLY IT TO EACH OBJECT
      return rectangle.length * rectangle.breadth;
    for (var i = 0;i<rectangleArray.length;i++) {</pre>
       var r = rectangleArray[i];
       console.log(r.length + "," + r.breadth + "," + getArea(r));
  printStuffAboutRectangleArray(rectArray);
  // the nested function getArea will not exist here!
```

```
this.lengtACTUALLY, WE DO EVEN BETTER - WE BUILD A
  this break ESTED FUNCTION INSIDE A NESTED FUNCTION!!
window.onload = function(){
  var rectangle1 = new Rectangle(3,4);
  var rectangle2 = new Rectangle(4,5);
 var rectangle3 = new OUTER NESTED FUNCTION, APPLY IT
 var rectArray = [rectangle1_rectangle2_rectangle2_rectangle3]r OBJECTS
//printStuffAboutRectangleArro (ANAAKKAY OF OBJECTS
 var printStuffAboutRectangleArray = function(rectangleArray)
    var getArea = function(rectangle){
                                                          tes the area of a
   INNER NESTED FUNCTION, APPLY IT TO EACH OBJECT
      return rectangle.length * rectangle.breadth;
    for (var i = 0;i<rectangleArray.length;i++) {</pre>
       var r = rectangleArray[i];
       console.log(r.length + "," + r.breadth + "," + getArea(r));
  printStuffAboutRectangleArray(rectArray);
  // the nested function getArea will not exist here!
```

```
LET'S MAKE SURE WE GOT THE BIG APPLY IT
                     PICTURE RIGHT!
var printStuffAboutRectangleArray = function(rectangleArray)
 var getArea = function(rectangle){
   console.log(
   return rectangle length * rectangle breadth; INNER NESTEP FUNCTION,
```

```
function Rectangle(l,b)
                        WE HAVE A SIMPLE OBJECT
  this.length = l;
  this.breadth = b;
                                 CONSTRUCTOR
window.onload = funttion(){
  var rectangle1 = new Rectangle(3,4);
  var rectangle2 = new Rectangle(4,5);
  var rectangle3 = new Rectangle(5,6);
  var rectArray = [rectangle1, rectangle2, rectangle3];
  //printStuffAboutRectangleArray(rectArray);
  var printStuffAboutRectangleArray = function(rectangleArray) {
    var getArea = function(rectangle){
      console.log("Inside a nested function that calculates the area of a
rectangle");
      return rectangle.length * rectangle.breadth;
    for (var i = 0;i<rectangleArray.length;i++) {</pre>
       var r = rectangleArray[i];
       console.log(r.length + "," + r.breadth + "," + getArea(r));
  printStuffAboutRectangleArray(rectArray);
  // the nested function getArea will not exist here!
```

```
function Rectangle(l,b) {
  this.length = l;
  this.breadth = b;
window.onload = funcWE(CREATE A FEW DIFFERENT OBJECTS
 var rectangle1 = new rectangle(1,1)$ OBJECT CONSTRUCTOR var rectangle2 = new Rectangle(4,1)$
  var rectangle3 = new Rectangle(5,6);
  var rectArray = [rectangle1, rectangle2, rectangle3];
  THERE IS SOME SIMPLE OPERATION WE'D ...
    LIKE TO PERFORM WITH ALL OF THESE.
      console.log("Inside a nested function that calculates the area of a
rectangle");
      return rectangle.length * rectangle.breadth;
    for (var i = 0;i<rectangleArray.length;i++) {</pre>
      var r = rectangleArray[i];
      console.log(r.length + "," + r.breadth + "," + getArea(r));
  printStuffAboutRectangleArray(rectArray);
  // the nested function getArea will not exist here!
```

```
function Rectangle (lab) [FATED AN ARRAY OF this breadth = h: CKEATED AN ARRAY OF
                       THESE OBJECTS..
window.onload = function
  var rectangle1 = new Rectangle(3,4);
  var rectangle2 = new Rectangle(4,5);
  var rectangle3 = new Rectangle(5,6);
  var rectArray = [rectangle1, rectangle2, rectangle3];
  //printStuffAboutRectangleArray(rectArray)
  var printStuffAboutRectangleArray = function(rectangleArray) {
    var getArea = function(rectangle){
      console.log("Inside a nested function that calculates the area of a
rectangle");
      return rectangle.length * rectangle.breadth;
    for (var i = 0;i<rectangleArray.length;i++) {</pre>
       var r = rectangleArray[i];
       console.log(r.length + "," + r.breadth + "," + getArea(r));
  printStuffAboutRectangleArray(rectArray);
  // the nested function getArea will not exist here!
```

```
function RectWE CREATED AN ARRAY OF THESE this.length WE CREATED AN ARRAY OF THESE
  this.breadth = b;
                                OBJECTS...
window.onload = function(){
  var rectangle1 = new Rectangle(3,4);
  var rectangle2 = new Rectangle(4,5);
 var rectangle3 = newAND PASSED IT TO A NESTED FUNCTION
  var rectArray = [rectangleTHE OUTER NESTED FUNCTION]
//printStuffAboutRectangleTHE OUTER NESTED FUNCTION]
  var printStuffAboutRectangleArray = function(rectangleArray) {
    var getArea = function(rectangle){
      console.log("Inside a nested function that calculates the area of a
rectangle");
      return rectangle.length * rectangle.breadth;
    for (var i = 0;i<rectangleArray.length;i++) {</pre>
       var r = rectangleArray[i];
       console.log(r.length + "," + r.breadth + "," + getArea(r));
  printStuffAboutRectangleArray(rectArray);
  // the nested function getArea will not exist here!
```

```
function RectWE CREATED AN ARRAY OF THESE this.length WE CREATED AN ARRAY OF THESE
  this.breadth = b;
                               OBJECTS..
window.onload = function(){
  var rectangle1 = new Rectangle(3,4);
  var rectangle2 = new Rectangle(4,5);
 var rectangle3 = newAND PASSED IT TO A NESTED FUNCTION
  var rectArray = [rectangleTHE OUTER NESTED FUNCTION]
//printStuffAboutRectangleTHE OUTER NESTED FUNCTION]
 var printStuffAboutRectangleArray = function(rectangleArray)
    var getArea = function(rectangle){
      console.log("Inside a nested function that calculates the area of a
rectangle");
      return rectangle.length * rectangle.breadth;
           AND CREATED AN INNER NESTED FUNCTION THAT
       var r OPERATES ON 1 ELEMENT OF THIS ARRAY!
  printStuffAboutRectangleArray(rectArray);
  // the nested function getArea will not exist here!
```

```
function Rectangle(l,b) {
 this.length = 1;
 this.breadth = b;
window.onload = function(){
  var rectangle1 = new Rectangle(3,4);
 var rectangle2 = new Rectangle(4,5);
  var rectangle3 = new Rectangle(5,6);
 l. CREATE ARRAY
var rectArray = [rectangle1, rectangle2, rectangle3];
 var printStuffAboutRectangleArray
function(rectangleArray) {
   var getArea = function(rectangle){
                                        2. CREATE (OUTER) NESTED
     console.log("Inside a nested function
rectangle");
    3. CALL NESTED FUNCTION rectangle. breadth;
   for WITH OUR ARRAYLeArray.length; i++) {
      console.log(r.length + "," + r.breadth + "," + getArea(r));
  printStuffAboutRectangleArray(rectArray);
  // the nested function getArea will not exist here!
  //getArea(rectangle1);
```

A FEW POINTS WORTH NOTING ABOUT THESE NESTED FUNCTIONS

- THESE ARE FUNCTION LITERALS AND SO ARE ONLY AVAILABLE AFTER THE FUNCTION EXPRESSION IS EVALUATED
 - THE INNER NESTED FUNCTION IS LOCAL TO THE OUTER NESTED FUNCTION AND CAN'T BE USED OUTSIDE IT

printStuffAboutRectangleArray(rectArray);

```
function Rectangle(l,b) {
this.lengt| Function Literals and so are only
  this Available after the function expression is evaluated
window.onload = function(){
  var rectangle1 = new Rectangle(3,4);
  var rectangle2 = new Rectangle(4,5);
  var rectangle3 = new Rectangle(5,6);
  var rectArray = [rectangle1, rectangle2, rectangle3];
  //printStuffAboutRectangleArray(rectArray);
  var printStuffAboutRectangleArray =
function(rectangleArray)
   console.log(HE FUNCTION EXPRESSION ISLEVALUATED HERE a
rectangle");
      return rectangle.length * rectangle.breadth;
   for (var i = 0;i<rectangleArray.length;i++) {</pre>
      var r = rectangleArray[i];
      console.log(r.length + "," + r.breadth + "," + getArea(r));
```

```
function Rectangle(l,b) {
this.lengt| Function Literals and so are only
  this Available after the function expression is evaluated
window.onload = function(){
  var rectangle1 = new Rectangle(3,4);
  var rectangle2 = new Rectangle(4,5);
  var rectangle3 = new Rectangle(5,6);
  var rectArray = [rectangle1, rectangle2, rectangle3];
  //printStuffAlteMREINGTOLLAFITAHERECMANULTYYJELD AN ERROR
  var printStuffAboutRectangleArray =
function(rectangleArray)
    var getArea = function(rectangle){
      console.log("Inside a nested function that calculates the area of a
rectangle");
      return rectangle.length * rectangle.breadth;
   for (var i = 0;i<rectangleArray.length;i++) {</pre>
      var r = rectangleArray[i];
       console.log(r.length + "," + r.breadth + "," + getArea(r));
```

A FEW POINTS WORTH NOTING ABOUT THESE NESTED FUNCTIONS

- THESE ARE FUNCTION LITERALS AND SO ARE ONLY AVAILABLE AFTER THE FUNCTION EXPRESSION IS EVALUATED
 - THE INNER NESTED FUNCTION IS LOCAL TO THE OUTER NESTED FUNCTION AND CAN'T BE USED OUTSIDE IT

printStuffAboutRectangleArray(rectArray);

• THE INNER NESTED FUNCTION IS LOCAL TO THE OUTER NESTED FUNCTION AND CAN'T BE USED OUTSIDE IT

```
OUTER NESTED FUNCTION
    var printStuffAboutRectangleArray = function(rectangleArray) {
    var getArea = function(rectangle){
       console.log("Inside a nested function that calculates the area of a
 rectangle");
       return rectangle.length * rectangle breadther FUNCTION
     for (var i = 0;i<rectangleArray.length;i++) {</pre>
        var r = rectangleArray[i];
        console.log(r.length + ',
  printStuffAboutRectangleArray(rectArray);
  // the nested function getArea will not exist here!
  //getArea(rectangle1);
```

var rethe inner nested function is local to the outer //prints nested function and can't be used outside it

```
var getArea = function(rectangle){
     console.log("Inside a nested function that calculates the area of a
rectangle");
      return rectangle.length * rectangle.breadth;
   for (var i = 0;i<rectangleArray.length;i++) {</pre>
         r r - rectangleArray[i].
       Uncaught ReferenceError: getArea is not defined
 printStuffAboutRectangleArray(rectArray);
   // the nested function getArea will not
  ist hereattempting to use the inner nester function //getArea(rectarg would yield an error
```

NESTED FUNCTIONS ARE KEY TO UNDERSTANDING AND USING CLOSURES

THIS MAKES THEM ONE OF THE MOST IMPORTANT CONCEPTS IN JAVASCRIPT.