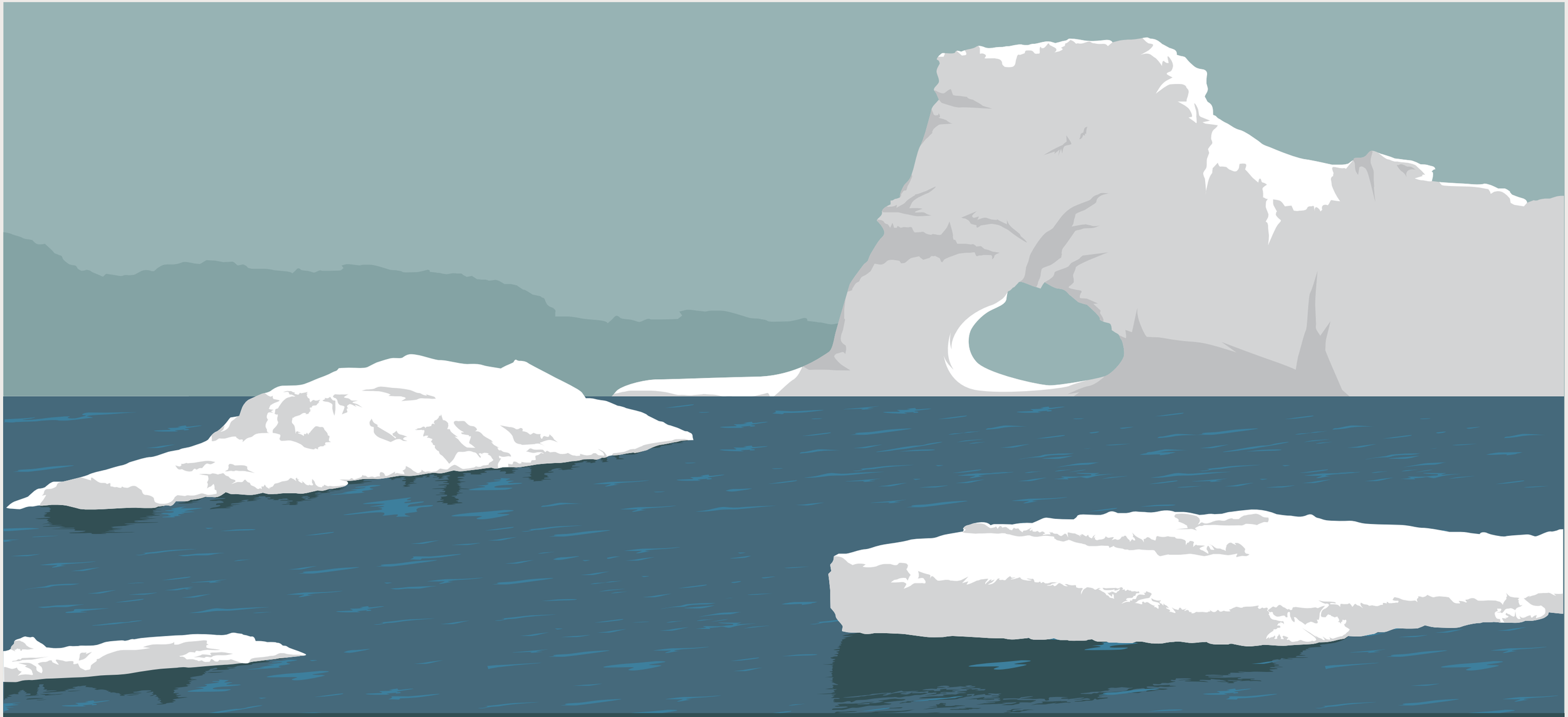


Explore

COLD CLOSURES COVE

LOOPS WITH CLOSURES: A CAUTIONARY TALE

Let's try to make a torpedo assigner for the Cove's Submarine

```
function assignTorpedo ( name, passengerArray ){



}
```

We'll pass in the name of a passenger, as well as a list of passengers.

LOOPS WITH CLOSURES: A CAUTIONARY TALE

Let's try to make a torpedo assigner for the Cove's Submarine

```
function assignTorpedo ( name, passengerArray ){
        var torpedoAssignment;

            This variable will hold a function that alerts name's torpedo assignment.




}
```

LOOPS WITH CLOSURES: A CAUTIONARY TALE

Let's try to make a torpedo assigner for the Cove's Submarine

```
function assignTorpedo ( name, passengerArray ){
        var torpedoAssignment;
        for (var i = 0; i<passengerArray.length; i++) {



        }

}
```

We'll loop over the list of passengers to find name.

LOOPS WITH CLOSURES: A CAUTIONARY TALE

Let's try to make a torpedo assigner for the Cove's Submarine

```
function assignTorpedo ( name, passengerArray ){
        var torpedoAssignment;
        for (var i = 0; i<passengerArray.length; i++) {
            if (passengerArray[i] == name) {
              torpedoAssignment = function ( ) {


              };
            }
        }

}
```

When we find the right name, we'll make a function that will hold our torpedo assignment closure.

LOOPS WITH CLOSURES: A CAUTIONARY TALE

Let's try to make a torpedo assigner for the Cove's Submarine

```
function assignTorpedo ( name, passengerArray ){
        var torpedoAssignment;
        for (var i = 0; i<passengerArray.length; i++) {
            if (passengerArray[i] == name) {
              torpedoAssignment = function ( ) {
                 alert("Ahoy, " + name + "!\n" +
                       "Man your post at Torpedo #" + (i+1) + "!");
              };
            }
        }

}
```

We'll close up the name variable and the loop counter i, and assign a person to the torpedo associated with their index value in the list (adjusted for zero).

LOOPS WITH CLOSURES: A CAUTIONARY TALE

Let's try to make a torpedo assigner for the Cove's Submarine

```
function assignTorpedo ( name, passengerArray ){
        var torpedoAssignment;
        for (var i = 0; i<passengerArray.length; i++) {
            if (passengerArray[i] == name) {
              torpedoAssignment = function ( ) {
                 alert("Ahoy, " + name + "!\n" +
                       "Man your post at Torpedo #" + (i+1) + "!");
              };
            }
        }
        return torpedoAssignment;
}
```

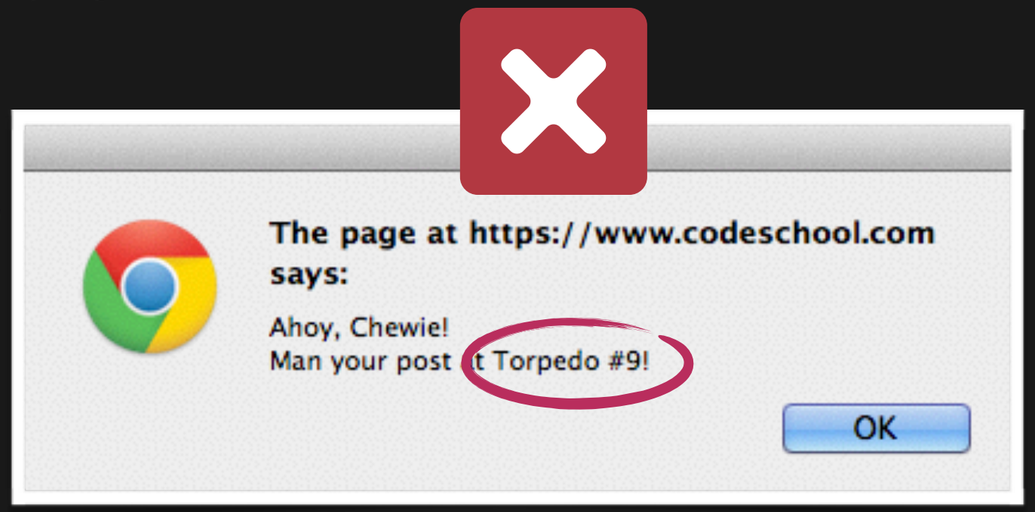
Finally, we'll hand the correct assignment back over to the global scope.

LOOPS WITH CLOSURES: A CAUTIONARY TALE

Let's try to make a torpedo assigner for the Cove's Submarine

```
function assignTorpedo ( name, passengerArray ){
        var torpedoAssignment;
        for (var i = 0; i<passengerArray.length; i++) {
            if (passengerArray[i] == name) {
              torpedoAssignment = function ( ) {
                 alert("Ahoy, " + name + "!\n" +
                        "Man your post at Torpedo #"
              };
            }
        }
        return torpedoAssignment;
}
```

Should be Torpedo #4!
What happened?




```
var subPassengers = ["Luke", "Leia", "Han", "Chewie", "Yoda", "R2-D2", "C-3PO", "Boba"];
```

```
var giveAssignment = assignTorpedo("Chewie", subPassengers);
```

```
giveAssignment();
```

CLOSURES BIND VALUES AT THE VERY LAST MOMENT

We have to pay close attention to return times and final variable states

```
function assignTorpedo ( name, passengerArray ){
        var torpedoAssignment;
        for (var i = 0; i<passengerArray.length; i++) {
            if (passengerArray[i] == name) {
              torpedoAssignment = function ( ) {
                 alert("Ahoy, " + name + "!\n" +
                       "Man your post at Torpedo #" + (i+1) + "!");
              };
            }
        }
        return torpedoAssignment;
}
```

Way before torpedoAssignment is returned, the i loop counter has progressed in value to 8 and stopped the loop.

```
var subPassengers = ["Luke", "Leia", "Han", "Chewie", "Yoda", "R2-D2", "C-3PO", "Boba"];
```

```
var giveAssignment = assignTorpedo("Chewie", subPassengers);
```

```
giveAssignment();
```

CLOSURES BIND VALUES AT THE VERY LAST MOMENT

We have to pay close attention to return times and final variable states

```
function assignTorpedo ( name, passengerArray ){
        var torpedoAssignment;
        for (var i = 0; i<passengerArray.length; i++) {
            if (passengerArray[i] == name) {
              torpedoAssignment = function ( ) {
                 alert("Ahoy, " + name + "!\n" +
                        "Man your post at Torpedo #" + (i+1) + "!");
              };
            }
        }
        return torpedoAssignment;
}
```

8+1=9

The function's actual return is the true "moment of closure," when the environment and all necessary variables are packaged up.

```
var subPassengers = ["Luke", "Leia", "Han", "Chewie", "Yoda", "R2-D2", "C-3PO", "Boba"];
```

```
var giveAssignment = assignTorpedo("Chewie", subPassengers);
```

```
giveAssignment();
```

WHAT CAN WE DO TO ENSURE THE CORRECT VALUE?

Several options exist for timing closures correctly

```
function assignTorpedo ( name, passengerArray ){
        var torpedoAssignment;
        for (var i = 0; i<passengerArray.length; i++) {
            if (passengerArray[i] == name) {
              torpedoAssignment = function ( ) {
                 alert("Ahoy, " + name + "!\n" +
                       "Man your post at Torpedo #" + (i+1) + "!");
              };
            }
        }
        return torpedoAssignment;
}
```

```
var subPassengers = ["Luke", "Leia", "Han", "Chewie", "Yoda", "R2-D2", "C-3PO", "Boba"];
```

WHAT CAN WE DO TO ENSURE THE CORRECT VALUE?

Several options exist for timing closures correctly

```
function assignTorpedo ( name, passengerArray ){

        for (var i = 0; i<passengerArray.length; i++) {
            if (passengerArray[i] == name) {
                                    function ( ) {
                alert("Ahoy, " + name + "!\n" +
                      "Man your post at Torpedo #" + (i+1) + "!");
              };
            }
        }

}
```

```
var subPassengers = ["Luke", "Leia", "Han", "Chewie", "Yoda", "R2-D2", "C-3PO", "Boba"];
```

WHAT CAN WE DO TO ENSURE THE CORRECT VALUE?

Several options exist for timing closures correctly

```
function assignTorpedo ( name, passengerArray ){

        for (var i = 0; i<passengerArray.length; i++) {
            if (passengerArray[i] == name) {
                return function ( ) {
                alert("Ahoy, " + name + "!\n" +
                      "Man your post at Torpedo #" + (i+1) + "!");
                };
            }
        }
}
```

Now the function will be immediately returned when the right name is found, locking i in place.

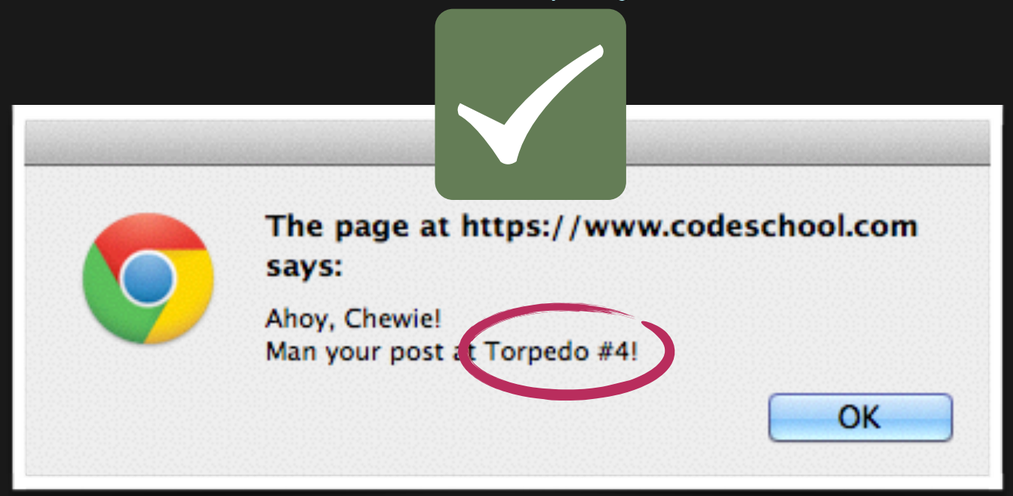
```
var subPassengers = ["Luke", "Leia", "Han", "Chewie", "Yoda", "R2-D2", "C-3PO", "Boba"];
```

WHAT CAN WE DO TO ENSURE THE CORRECT VALUE?

Several options exist for timing closures correctly

```
function assignTorpedo ( name, passengerArray ){

        for (var i = 0; i<passengerArray.length; i++) {
            if (passengerArray[i] == name) {
                return function ( ) {
                alert("Ahoy, " + name + "!\n" +
                      "Man your post at Torpedo #"
                };
            }
        }

}
```

An immediate return has the expected effect, because i is not allowed to progress!




```
var subPassengers = ["Luke", "Leia", "Han", "Chewie", "Yoda", "R2-D2", "C-3PO", "Boba"];
```

```
var giveAssignment = assignTorpedo("Chewie", subPassengers);
```

```
giveAssignment();
```

WHAT CAN WE DO TO ENSURE THE CORRECT VALUE?

We could also design the torpedo assigners a bit more like our ticket makers

```
function makeTorpedoAssigner ( passengerArray ) {

        return function ( name ) {



        };

}
```

This time our external function will only take in the passengerArray, and we'll let the returned function deal with a specific name.

```
var subPassengers = ["Luke", "Leia", "Han", "Chewie", "Yoda", "R2-D2", "C-3PO", "Boba"];
```

WHAT CAN WE DO TO ENSURE THE CORRECT VALUE?

We could also design the torpedo assigners a bit more like our ticket makers

```
function makeTorpedoAssigner ( passengerArray ) {

        return function ( name ) {
           for (var i = 0; i<passengerArray.length; i++) {


           }
        };

}
```

At this point, whatever passengerArray got passed in to makeTorpedoAssigner will be bound into the closure. Parameters are part of the environment, too!

```
var subPassengers = ["Luke", "Leia", "Han", "Chewie", "Yoda", "R2-D2", "C-3PO", "Boba"];
```

WHAT CAN WE DO TO ENSURE THE CORRECT VALUE?

We could also design the torpedo assigners a bit more like our ticket makers

```
function makeTorpedoAssigner ( passengerArray ) {

        return function ( name ) {
            for (var i = 0; i<passengerArray.length; i++) {
                if (passengerArray[i] == name) {
                    alert("Ahoy, " + name + "!\n" +
                          "Man your post at Torpedo #" + (i+1) + "!");
                }
            }
        };

}
```

Since we've put the loop inside the returned function, i will come directly from that local scope.

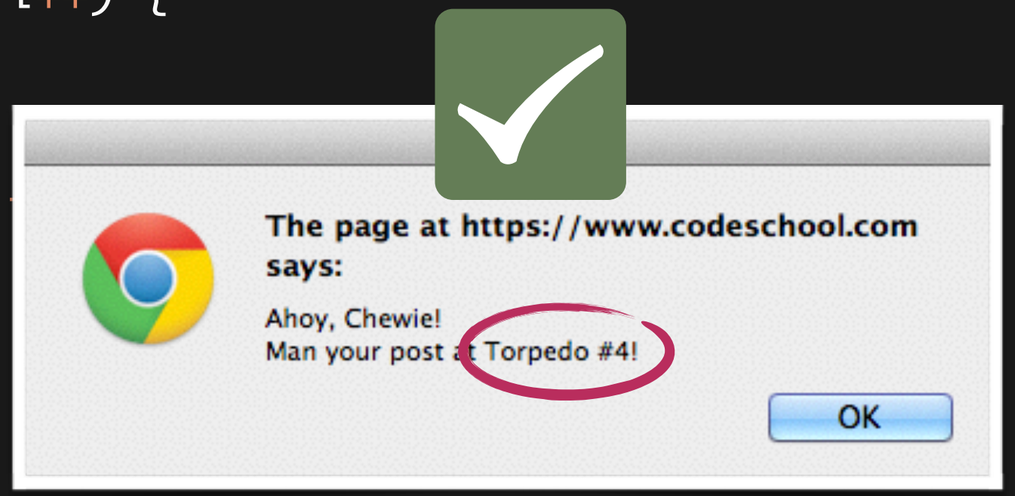
The only closed variable from the external scope is passengerArray, which never changes.

```
var subPassengers = ["Luke", "Leia", "Han", "Chewie", "Yoda", "R2-D2", "C-3PO", "Boba"];
```

NOW WE CAN PASS OUT TORPEDOES LIKE CANDY

TIE Fighter, dead ahead!...Er, underwater...

```
function makeTorpedoAssigner ( passengerArray ) {

        return function ( name ) {
           for (var i = 0; i<passengerArray.length; i++) {
              if (passengerArray[i] == name) {
                 alert("Ahoy, " + name + "!\n" +
                       "Man your post at Torpedo #"
              }
           }
        };

}
```




```
var subPassengers = ["Luke", "Leia", "Han", "Chewie", "Yoda", "R2-D2", "C-3PO", "Boba"];
```

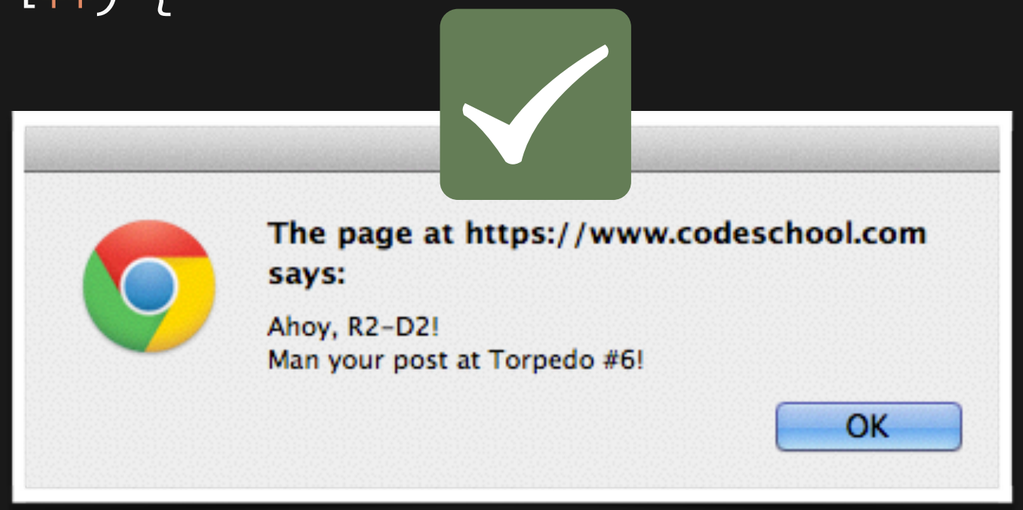
```
var getTorpedoFor = makeTorpedoAssigner(subPassengers);
```

```
getTorpedoFor("Chewie");
```

NOW WE CAN PASS OUT TORPEDOES LIKE CANDY

TIE Fighter, dead ahead!...Er, underwater...

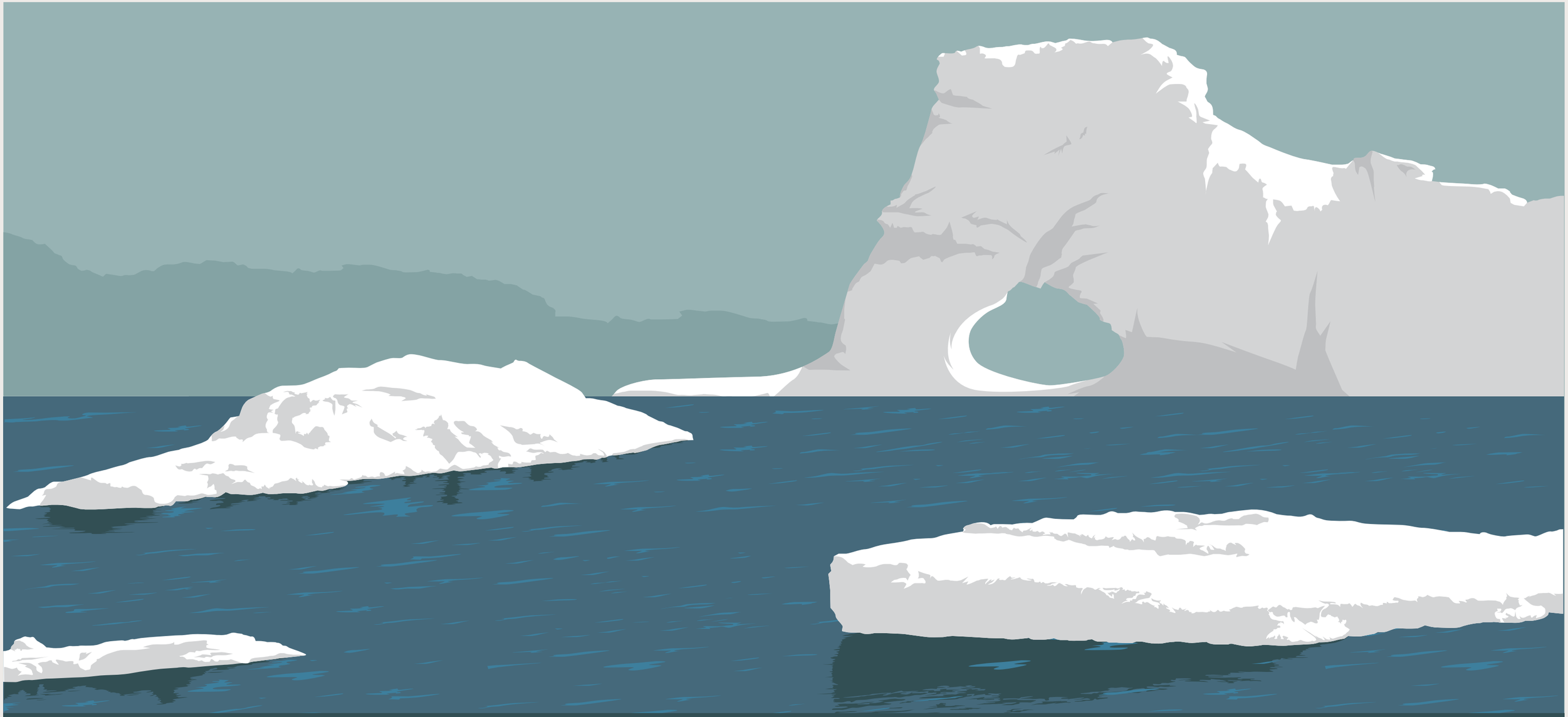
```
function makeTorpedoAssigner ( passengerArray ) {

        return function ( name ) {
           for (var i = 0; i<passengerArray.length; i++) {
              if (passengerArray[i] == name) {
                 alert("Ahoy, " + name + "!\n" +
                       "Man your post at Torpedo #"
              }
           }
        };

}
```




```
var subPassengers = ["Luke", "Leia", "Han", "Chewie", "Yoda", "R2-D2", “C-3PO”, “Boba”];
```

```
var getTorpedoFor = makeTorpedoAssigner(subPassengers);
```

```
getTorpedoFor( "R2-D2" );
```



Explore

COLD CLOSURES COVE