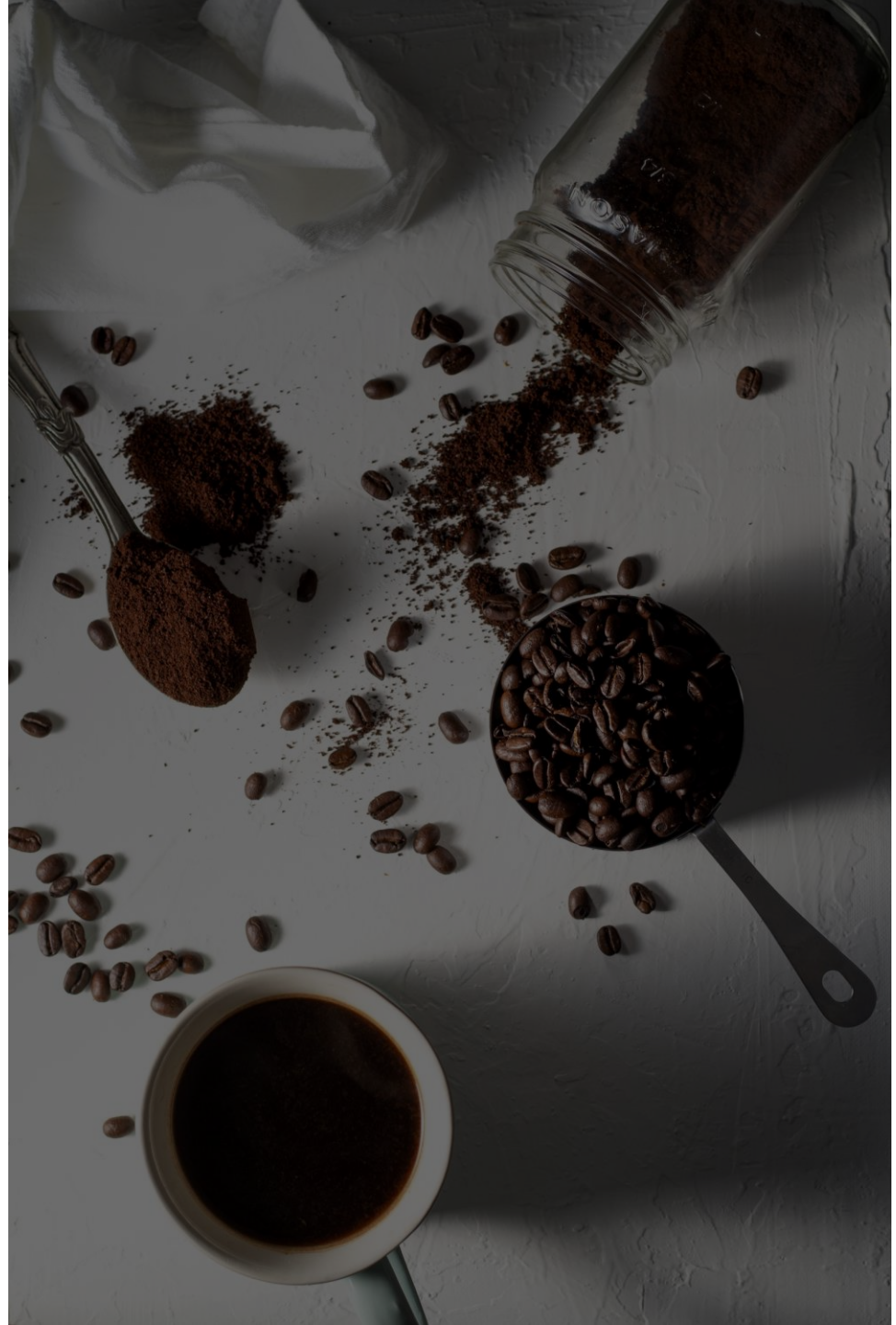


RENDER COFFEE LIST

THE HOW

```
var tbody = document.querySelector('#coffees');
tbody.innerHTML = renderCoffees(coffees);

function renderCoffees(coffees) {
  var html = '';
  for(var i = coffees.length - 1; i >= 0; i--) {
    html += renderCoffee(coffees[i]);
  }
  return html;
}

function renderCoffee(coffee) {
  var html = '<div class="coffee col-md-6 d-flex mb-3 p-2">';
  html += '<img src=' + coffee.imgURL + ' class="image" height="30px" mr-2 />';
  html += '<h3 id="name" class="ml-2">' + coffee.name + '</h3>';
  html += '<p>' + coffee.roast + '</p>';
  html += '</div>';
  return html;
}
```

The renderCoffees() function is doing two things:

1. Iterating through the coffee array
2. Perform a callback to the renderCoffee and assigns the result of each iteration as its argument, updating the html variable until false)
3. Returning the html variable (contains all the iterations of the for loop)

{id: 14, **name: French, roast: dark**}

```
<div class="coffee col-md-6 d-flex mb-3 p-2"><img src=undefined class="image" height="30px" mr-2 /><h3 id="name" class="ml-2">French</h3><p>dark</p></div>
```

Happens once each time the for loops runs

UPDATE THE DISPLAY COFFEE LIST – *OPTION DROP-DOWN INPUT*

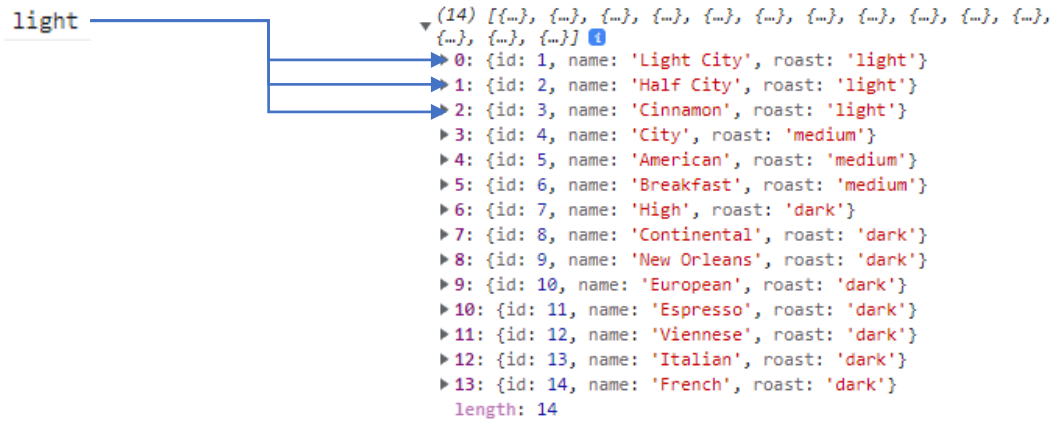
THE HOW

```
var roastSelection = document.querySelector('#roast-selection');


roastSelection.addEventListener("change", function() {
   let selectedRoast = roastSelection.value;
   let html = '';
   coffees.forEach(coffee => {
      if (selectedRoast === coffee.roast) {
         console.log(renderCoffee(coffee));
         tbody.innerHTML = html +=
renderCoffee(coffee);

      } else if (selectedRoast === "all") {
         tbody.innerHTML = renderCoffees(coffees);
      }
   })
})
```

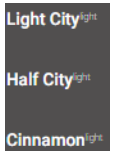
1. Place drop-down ID in a variable: roastSelection

2. Place roastSelection value (lets use ‘light’) in a variable selectedRoast (expected output: ‘light’)

3. Perform a forEach on coffees to determine if selectedRoast matches the roast value in coffee



3a. If it does we perform a callback to the renderCoffee and assigns the result of each iteration as its argument, updating the html variable until false)



3b. If it doesn’t we perform a callback to the renderCoffee function and update the tbody innerHTML.

UPDATE THE DISPLAY COFFEE LIST – *User input*

THE HOW

```
let search = document.querySelector("#searchCoffee");

search.addEventListener('keyup', searchCoffees)


function searchCoffees(e) {
   e.preventDefault();
   let coffeeName = search.value;
   var filteredCoffees = [];
   coffees.forEach(function(coffee) {
      if (coffee.name.toUpperCase().indexOf(coffeeName.toUpperCase()) > -1)
{filteredCoffees.push(coffee);
      }
   });
   console.log(filteredCoffees);
   tbody.innerHTML = renderCoffees(filteredCoffees)
}}
```

1. Place user input ID in a variable: **search**

2. Add an event listener to our search variable with a keyup event and the **searchCoffees** function.
   keyup - fires when you release a key on the keyboard.

3. preventDefault() - *This method gives the ability to prevent a browser's default behavior for events.*

4. Place search value (lets use 'f') in a variable **coffeeName** (expected output: 'f')

5. Perform a forEach on coffees to determine if **coffee.name** contains **coffeeName** ('f

console.log("coffee.name: ", coffee.name)

console.log("coffeeName: ",coffeeName)

5a. If it does we push **coffee** to **filteredCoffees**

console.log("coffee: ", coffee)

console.log('filteredCoffees', filteredCoffees);

5b. We then do a callback on renderCoffees(filteredCoffees) with the variable filteredCoffees as its argument and set it equal to tbody.innerHTML.

ADD AND DISPLAY THE COFFEE LIST – *User input*

THE HOW

```
let search = document.querySelector("#searchCoffee");

search.addEventListener('keyup', searchCoffees)


function searchCoffees(e) {
   e.preventDefault();
   let coffeeName = search.value;
   var filteredCoffees = [];
   coffees.forEach(function(coffee) {
      if (coffee.name.toUpperCase().indexOf(coffeeName.toUpperCase()) > -1)
{filteredCoffees.push(coffee);
      }
   });
   console.log(filteredCoffees);
   tbody.innerHTML = renderCoffees(filteredCoffees)
}}
```

1. Place user input ID in a variable: **search**

2. Add an event listener to our search variable with a keyup event and the **searchCoffees** function.
   keyup - fires when you release a key on the keyboard.

3. preventDefault() - *This method gives the ability to prevent a browser's default behavior for events.*

4. Place search value (lets use 'f') in a variable **coffeeName** (expected output: 'f')

5. Perform a forEach on coffees to determine if **coffee.name** contains **coffeeName** ('f

```
coffee.name:  Half City
coffee.name:  Breakfast
coffee.name:  French
```

```
coffeeName:  f
```

console.log("coffee.name: ", coffee.name)

console.log("coffeeName: ",coffeeName)

5a. If it does we push **coffee** to **filteredCoffees**

```
coffee:  ▸ {id: 2, name: 'Half City', roast: 'light'}
coffee:  ▸ {id: 6, name: 'Breakfast', roast: 'medium'}
coffee:  ▸ {id: 14, name: 'French', roast: 'dark'}
```

```
filteredCoffees
▾ (3) [{…}, {…}, {…}]
  ▸ 0: {id: 2, name: 'Half City', roast: 'light'}
  ▸ 1: {id: 6, name: 'Breakfast', roast: 'medium'}
  ▸ 2: {id: 14, name: 'French', roast: 'dark'}
    length: 3
```

console.log("coffee: ", coffee)

console.log('filteredCoffees', filteredCoffees);

5b. We then do a callback on renderCoffees(filteredCoffees) with the variable filteredCoffees as its argument and set it equal to tbody.innerHTML.