PROCESSING DATA FROM WEB

We have fetch the data in html form from cricinfo website under fixtures and results. But we do not need the matches information in html form we need to process information and put them in an excel file.

Javascript > Lecture 26 > JS FirstProcessingHTML.js > ...

```
1   // npm install jsdom
2   // node FirstProcessingHTML.js --source=download.html
3
4   let minimist = require("minimist");
5   let fs = require("fs");
6   let jsdom = require("jsdom"); // will load html and help find information
7
8   let args = minimist(process.argv);
9   console.log(args.source);
```

Now we have to process download.html file

download.html

PROBLEMS OUTPUT TERMINAL DEBUG CONSOLE

```
→  Lecture 26 git:(main) ✗ node FirstProcessingHTML.js --source=download.html
download.html
→  Lecture 26 git:(main) ✗
```

JS FirstProcessingHTML.js U

Javascript > Lecture 26 > JS FirstProcessingHTML.js > ...

```
1   // npm install jsdom
2   // node FirstProcessingHTML.js --source=download.html
3
4   let minimist = require("minimist");
5   let fs = require("fs");
6   let jsdom = require("jsdom"); // will load html and help find information
7
8   let args = minimist(process.argv);
9
10  fs.readFile(args.source, "utf-8", function(err, data){
11      console.log(data);
12  });
```

PROBLEMS OUTPUT TERMINAL DEBUG CONSOLE

```
        })();
      </script><script type="text/javascript">
        var _sf_async_config={};
        /** CONFIGURATION START **/
        _sf_async_config.uid = 26455;
```

JS FirstProcessingHTML.js U ×

Javascript > Lecture 26 > JS FirstProcessingHTML.js > fs.readFile("utf-8") callback

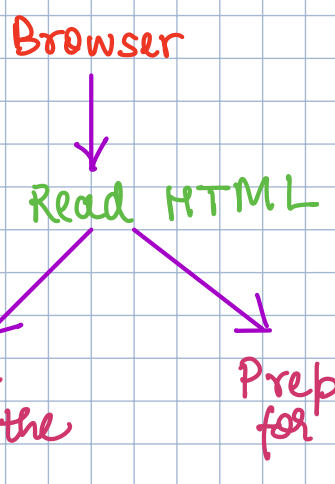
```
1   // npm install jsdom
2   // node FirstProcessingHTML.js --source=download.html
3
4   let minimist = require("minimist");
5   let fs = require("fs");
6   let jsdom = require("jsdom"); // will load html and help find information
7
8   let args = minimist(process.argv);
9
10  fs.readFile(args.source, "utf-8", function(err, html){
11      let JSDOM = jsdom.JSDOM;
12      let dom = new JSDOM(html);
13      let document = dom.window.document;
14  });
```

→ library jsdom get JSDOM out
← we passed html and it provides DOM

From server —we get→ html —provide to→ Browser

```
<html>
  <head>
  </head>
  <body>
    <p>
    <input type="button" />
  </body>
</html>
```

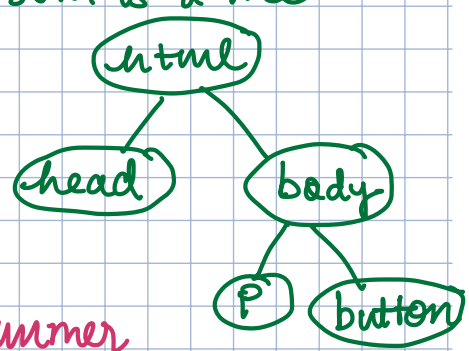
Browser has two responsibilities:-



Document Object Model

What is DOM?

DOM is a tree



→ Browser prepares DOM for the programmer, so that the programmer can take out information from this DOM tree.

sample.html U

Javascript > Lecture 26 > sample.html > html

```
1   <html>
2       <head>
3           <title>My sample page</title>
4       </head>
5       <body>
6           <p>Hello there</p>
7           <button>Awesome1</button>
8           <input type = "text" placeholder="Sample1">
9           <div>
10          <p>Goodbye</p>
11          <button>Awesome2</button>
12          <input type = "text" placeholder="Sample2">
13          </div>
14      </body>
15  </html>
```

inspect ↓

Hello there

Awesome1 Sample1

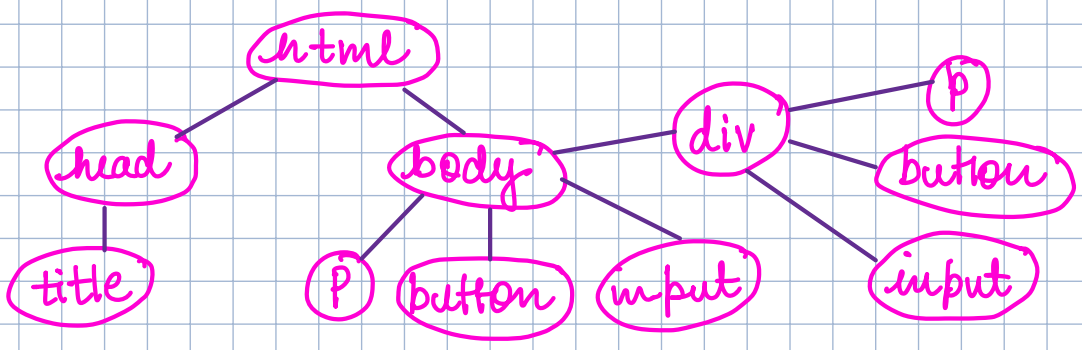
Goodbye

Awesome2 Sample2

Elements Console Sources Network

```
<html>
▶ <head>…</head>
▼ <body>
    <p>Hello there</p>
    <button>Awesome1</button>
    <input type="text" placeholder="Sample1">
  ▼ <div>
...   <p>Goodbye</p> == $0
      <button>Awesome2</button>
      <input type="text" placeholder="Sample2">
    </div>
  </body>
</html>
```

DOM
↓
Document
Object
Model



→ using DOM programmer can control browser

→ DOM is an object tree which is created by the browser for the programmer.

→ Programmer can interact with view of page using DOM.

Javascript > Lecture 26 > JS FirstProcessingHTML.js > fs.readFile("utf-8") callback

```
1   // npm install jsdom
2   // node FirstProcessingHTML.js --source=download.html
3
4   let minimist = require("minimist");
5   let fs = require("fs");
6   let jsdom = require("jsdom"); // will load html and prepare the dom
7                                 // for programmer just like a browser would have
8
9   let args = minimist(process.argv);
10
11  fs.readFile(args.source, "utf-8", function(err, html){
12      let JSDOM = jsdom.JSDOM;
13      let dom = new JSDOM(html);
14      let document = dom.window.document;
15
16      console.log(document.title);
17  });
```

PROBLEMS OUTPUT TERMINAL DEBUG CONSOLE

```
→  Lecture 26 git:(main) ✗ node FirstProcessingHTML.js --source=download.html
ICC Cricket World Cup - Cricket Schedules, Updates, Results | ESPNcricinfo.com
→  Lecture 26 git:(main) ✗
```

use of querySelector() and querySelectorAll()

FirstProcessingHTML.js U

Javascript > Lecture 26 > JS FirstProcessingHTML.js > fs.readFile("utf-8") callback

```
7                                 // for programmer just like a browser w
8
9   let args = minimist(process.argv);
10
11  fs.readFile(args.source, "utf-8", function(err, html){
12      let JSDOM = jsdom.JSDOM;
13      let dom = new JSDOM(html);
14      let document = dom.window.document;
15
16      let b2 = document.querySelector(".b");
17      console.log(b2.textContent);
18
19  });
```

sample.html U

vascript > Lecture 26 > sample.html > html > body > div >

```
1   <html>
2       <head>
3           <title>My sample page</title>
4       </head>
5       <body>
6           <p>Hello there</p>
7           <button class = "a">Awesome1</button>
8           <input type = "text" placeholder="Sample1">
9           <div>
10          <p>Goodbye</p>
11          <button class = "b">Awesome2</button>
12          <input type = "text" placeholder="Sample2">
13          </div>
14      </body>
15  </html>
```

PROBLEMS OUTPUT TERMINAL DEBUG CONSOLE zsh

```
→  Lecture 26 git:(main) ✗ node FirstProcessingHTML.js --source=sample.html
Awesome2
→  Lecture 26 git:(main) ✗
```

FirstProcessingHTML.js U

Javascript > Lecture 26 > FirstProcessingHTML.js > fs.readFile("utf-8") callback

```
                    // for programmer just like a browser w

let args = minimist(process.argv);

fs.readFile(args.source, "utf-8", function(err, html){
    let JSDOM = jsdom.JSDOM;
    let dom = new JSDOM(html);
    let document = dom.window.document;

    let btns = document.querySelectorAll("button");
    console.log(btns.length);
    console.log(btns[0].textContent);
    console.log(btns[1].textContent);


});
```

sample.html U

Javascript > Lecture 26 > sample.html > html > body > div >

```
<html>
    <head>
        <title>My sample page</title>
    </head>
    <body>
        <p>Hello there</p>
        <button class = "a">Awesome1</button>
        <input type = "text" placeholder="Sample1">
        <div>
        <p>Goodbye</p>
        <button class = "b">Awesome2</button>
        <input type = "text" placeholder="Sample2">
        </div>
    </body>
</html>
```

PROBLEMS OUTPUT TERMINAL DEBUG CONSOLE zsh

```
→ Lecture 26 git:(main) ✗ node FirstProcessingHTML.js --source=sample.html
2
Awesome1
Awesome2
→ Lecture 26 git:(main) ✗
```

FirstProcessingHTML.js U

Javascript > Lecture 26 > FirstProcessingHTML.js > fs.readFile("utf-8") callback

```
                    // for programmer just like a browser w

let args = minimist(process.argv);

fs.readFile(args.source, "utf-8", function(err, html){
    let JSDOM = jsdom.JSDOM;
    let dom = new JSDOM(html);
    let document = dom.window.document;

    let elements = document.querySelectorAll(".b");
    console.log(elements.length);
    console.log(elements[0].textContent);
    console.log(elements[1].textContent);
    console.log(elements[2].textContent);

});
```

sample.html U

Javascript > Lecture 26 > sample.html > html > body > bu

```
<html>
    <head>
        <title>My sample page</title>
    </head>
    <body>
        <p class = "b">Hello there</p>
        <button class = "b">Awesome1</button>
        <input type = "text" placeholder="Sample1">
        <div>
        <p>Goodbye</p>
        <button class = "b">Awesome2</button>
        <input type = "text" placeholder="Sample2">
        </div>
    </body>
</html>
```

PROBLEMS OUTPUT TERMINAL DEBUG CONSOLE zsh

```
→ Lecture 26 git:(main) ✗ node FirstProcessingHTML.js --source=sample.html
3
Hello there
Awesome1
Awesome2
→ Lecture 26 git:(main) ✗
```

→ querySelector() and querySelectorAll() are two Javascript functions very useful when working with HTML elements and Javascript. With these functions you can get in Javascript, the HTML elements according to a group of CSS selectors ('id', 'class').

querySelector(): returns first element within the document that matches with specified group of selectors or null if no matches are found.

Syntax:

```
var elm = document.querySelector('selectors');
```

— 'selectors' is a string containing one or more CSS selectors separated by commas

— elm is an element object.

querySelectorAll() : returns an array of objects with the elements that match the specified group of selectors.

Syntax :

```
var elms = document.querySelectorAll('selectors');
```

- 'selector' is a string containing one or more CSS selectors by commas.
- elms is an array with selected HTML elements.

The difference between querySelector() and querySelectorAll() is that querySelector() returns a single object with the first HTML element that matches the "selectors" but querySelectorAll() returns an array of objects with all the HTML elements that match the "selectors".

https://coursesweb.net/javascript/queryselector-queryselectorall

FirstProcessingHTML.js U

Javascript > Lecture 26 > FirstProcessingHTML.js > fs.readFile("utf-8") callback

```
8
9   let args = minimist(process.argv);
10
11  fs.readFile(args.source, "utf-8", function(err, html){
12      let JSDOM = jsdom.JSDOM;
13      let dom = new JSDOM(html);  // constructor
14      let document = dom.window.document;
15
16      let elements = document.querySelectorAll("button.b");
17      console.log(elements.length);
18      console.log(elements[0].textContent);
19      console.log(elements[1].textContent);
20
21  });
```

only button element with class='b'

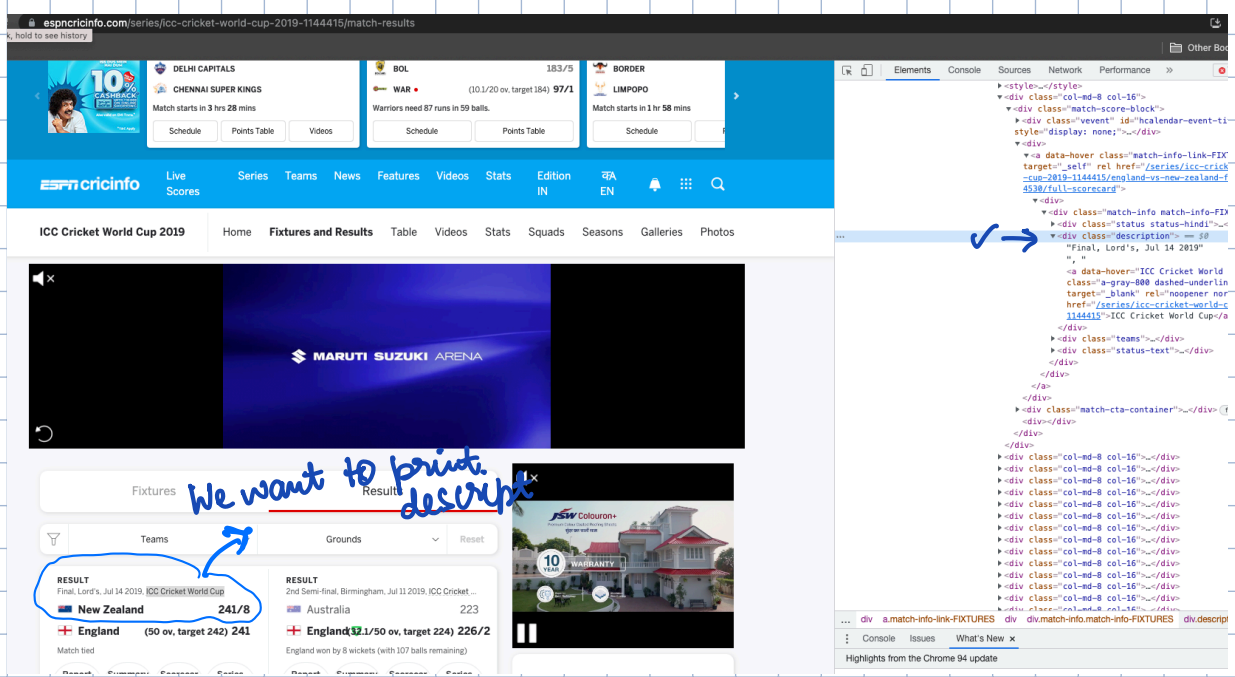
sample.html U

Javascript > Lecture 26 > sample.html > html > body > button.b

```
1   <html>
2       <head>
3           <title>My sample page</title>
4       </head>
5       <body>
6           <p class = "b">Hello there</p>
7           <button class = "b">Awesome1</button>
8           <input type = "text" placeholder="Sample1">
9           <div>
10          <p>Goodbye</p>
11          <button class = "b">Awesome2</button>
12          <input type = "text" placeholder="Sample2">
13          </div>
14      </body>
15  </html>
```

PROBLEMS OUTPUT TERMINAL DEBUG CONSOLE zsh

```
→  Lecture 26 git:(main) ✗ node FirstProcessingHTML.js --source=sample.html
2
Awesome1
Awesome2
→  Lecture 26 git:(main) ✗
```



inspect cric-info results page

JS FirstProcessingHTML.js U

Javascript > Lecture 26 > JS FirstProcessingHTML.js > ...

```
1   // npm install jsdom
2   // node FirstProcessingHTML.js --source=download.html
3
4   let minimist = require("minimist");
5   let fs = require("fs");
6   let jsdom = require("jsdom"); // will load html and prepare the dom
7                                 // for programmer just like a browser would have
8
9   let args = minimist(process.argv);
10
11  fs.readFile(args.source, "utf-8", function(err, html){
12      let JSDOM = jsdom.JSDOM;
13      let dom = new JSDOM(html);  // constructor
14      let document = dom.window.document;
15
16      let descs = document.querySelectorAll("div.description");
17      for(let i = 0; i< descs.length; i++){
18          console.log(descs[i].textContent);
19      }
20
```

we will get all the div elements that contains class description

PROBLEMS OUTPUT TERMINAL DEBUG CONSOLE

```
 Check Australia vs England 2nd Semi-final Videos, Reports Articles Online.
2nd Semi-final, Birmingham, Jul 11 2019, ICC Cricket World Cup
Check New Zealand vs India, ICC Cricket World Cup 2019, 1st Semi-final Match Timings, sc
 Check New Zealand vs India 1st Semi-final Videos, Reports Articles Online.
1st Semi-final, Manchester, Jul 9 - 10 2019, ICC Cricket World Cup
```

JS FirstProcessingHTML.js U

Javascript > Lecture 26 > JS FirstProcessingHTML.js > fs.readFile("utf-8") callback

```
1   // npm install jsdom
2   // node FirstProcessingHTML.js --source=download.html
3
4   let minimist = require("minimist");
5   let fs = require("fs");
6   let jsdom = require("jsdom"); // will load html and prepare the dom
7                                 // for programmer just like a browser would have
8
9   let args = minimist(process.argv);
10
11  fs.readFile(args.source, "utf-8", function(err, html){
12      let JSDOM = jsdom.JSDOM;
13      let dom = new JSDOM(html);  // constructor
14      let document = dom.window.document;
15
16      let descs = document.querySelectorAll("div.match-info > div.description");
17      for(let i = 0; i< descs.length; i++){
18          console.log(descs[i].textContent);
19      }
20  });
```

we will get div's with class description whose parent is a div with class match-info

we will get all the div elements that contains class description and that has a parent div element that has class match info

PROBLEMS OUTPUT TERMINAL DEBUG CONSOLE

```
Final, Lord's, Jul 14 2019, ICC Cricket World Cup
2nd Semi-final, Birmingham, Jul 11 2019, ICC Cricket World Cup
1st Semi-final, Manchester, Jul 9 - 10 2019, ICC Cricket World Cup
45th match (D/N), Manchester, Jul 6 2019, ICC Cricket World Cup
44th match, Leeds, Jul 6 2019, ICC Cricket World Cup
43rd match, Lord's, Jul 5 2019, ICC Cricket World Cup
42nd match, Leeds, Jul 4 2019, ICC Cricket World Cup
41st match, Chester-le-Street, Jul 3 2019, ICC Cricket World Cup
40th match, Birmingham, Jul 2 2019, ICC Cricket World Cup
39th match, Chester-le-Street, Jul 1 2019, ICC Cricket World Cup
38th match, Birmingham, Jun 30 2019, ICC Cricket World Cup
```