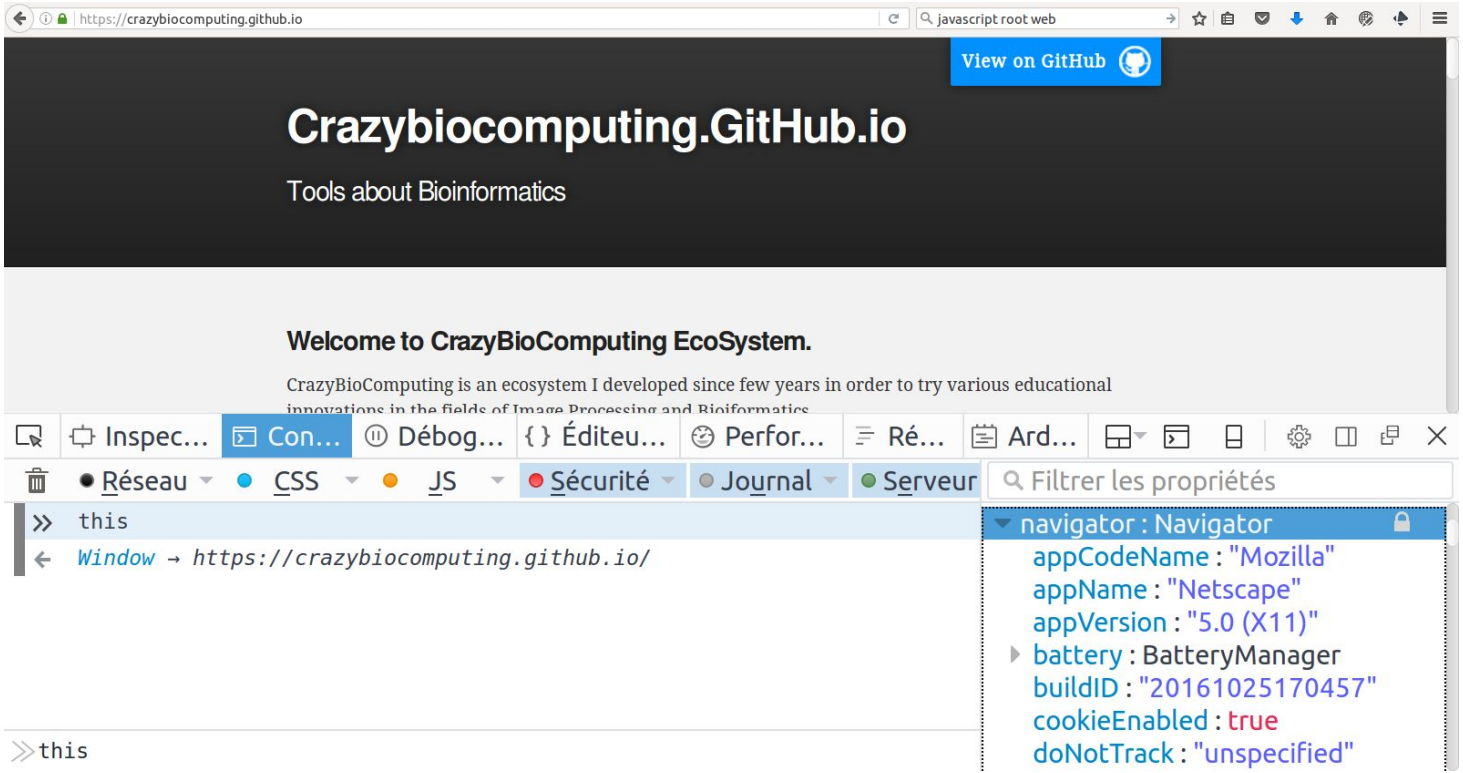
Web Technologies & Image Processing

Jean-Christophe Taveau

2016-2017

JS: introduction

JS and web browser



JS: Introduction

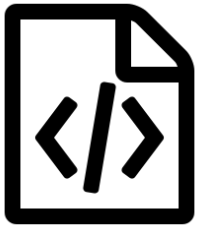
JS + web browser

```
Window
    +-- navigator
    +-- document
        +-- head
            +-- ...
        +-- body
            +-- ...
            +-- ...
```

```
>> navigator
Navigator {
  permissions   : Permissions,
  mimeTypes     : MimeTypeArray,
  plugins       : PluginArray,
  battery       : BatteryManager,
  oscpu         : "Linux x86_64",
  productSub    : "20100101",
  cookieEnabled: true
}
```

```
>> document.head.children
HTMLCollection [
  <meta>,
  <meta>,
  <meta>,
  <link>,
  <title>
]
```

JavaScript and Document Object Model



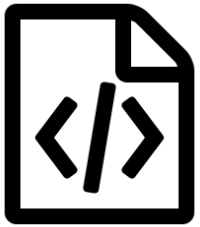
Root: document.documentElement

→ read-only property returns the root element of the document

→ for example, the <html> element for HTML documents.

JS: Document Object Model

JavaScript and the DOM

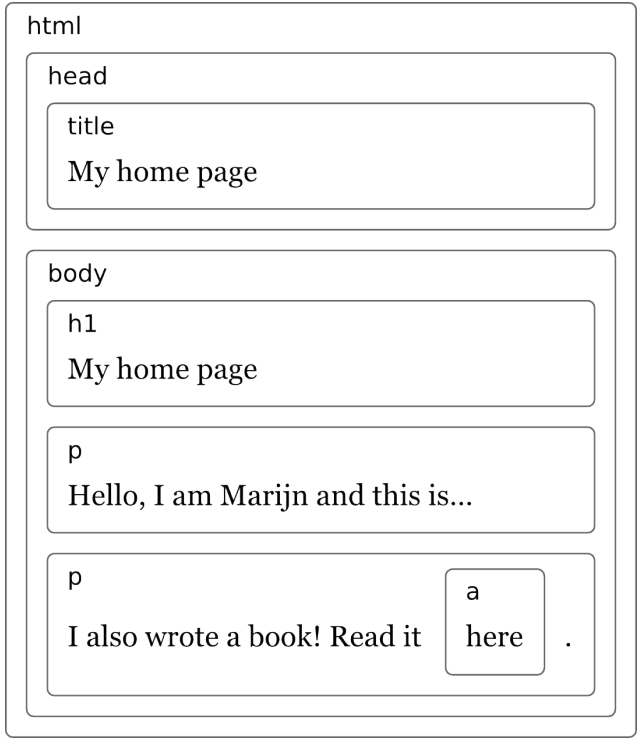


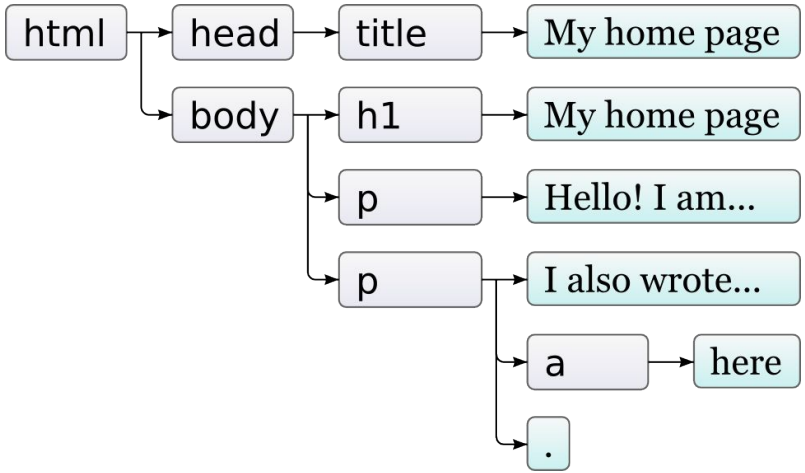
```
<!doctype html>
<html>
  <head>
    <title>My home page</title>
  </head>
  <body>
    <h1>My home page</h1>
    <p>Hello, I am Marijn and this is my home page.</p>
    <p>I also wrote a book! Read it
    <a href="http://eloquentjavascript.net">here</a>.</p>
  </body>
</html>
```

Source: http://eloquentjavascript.net/12_browser.html

JS: Document Object Model

JavaScript and the DOM

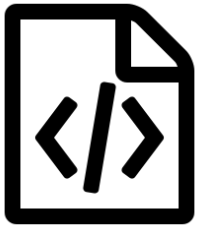




Source: http://eloquentjavascript.net/12_browser.html

JS: Document Object Model

JavaScript and the DOM



document The web page.

element An *element* refers to an element or a node of type element returned by a member of the DOM API.

nodeList A nodeList is an array of elements.
Items in a nodeList are accessed by index in either of two ways:

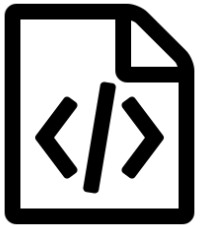
- list.item(1)
- list[1]

attribute An attribute of a HTML element.

Source: https://developer.mozilla.org/en-US/docs/Web/API/Document_Object_Model/Introduction

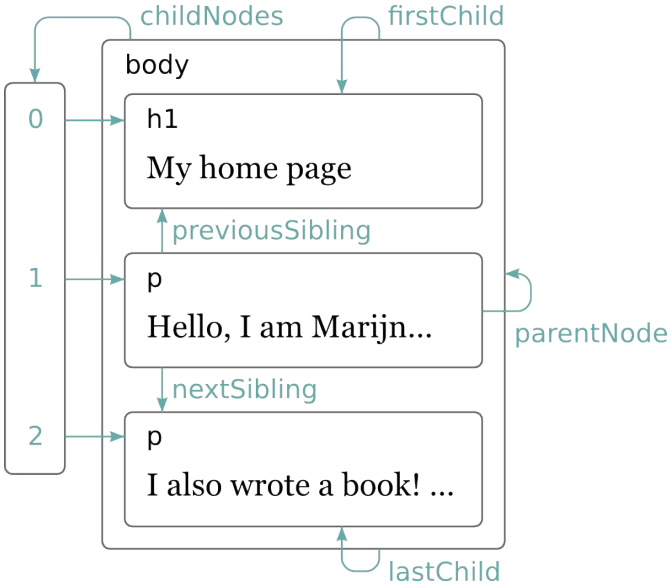
JS: Document Object Model

JavaScript and the DOM



Properties of document

```
document.body.children;
document.body.firstChild;
document.body.lastChild;
```



Source: http://eloquentjavascript.net/12_browser.html

JS: Finding Elements

Finding elements

document.getElementById(String id)
Returns an object reference to the identified element.

document.getElementsByClassName(String classname)
Returns a list of elements with the given class name.

document.getElementsByTagName(String tagname)
Returns a list of elements with the given tag name.

document.querySelector(String selector)
Returns the first Element node within the document, in document order, that matches the specified selectors.

JS: Finding Elements

```
var link = document.body.getElementsByTagName("a")[0];
console.log(link.href);
```

```
<p>My ostrich Gertrude:</p>
<p><img id="gertrude" src="img/ostrich.png"></p>

<script>
  var ostrich = document.getElementById("gertrude");
  console.log(ostrich.src);
</script>
```

JS: Creating Elements

document.createElement(String id)
Creates a new element with the given tag name.

Node.appendChild(Node childNode**)**
Adds the specified childNode argument as the last child to the current node.

```
<div id=”new”></div>

<script>
  var content = document.getElementById("new");
  var el = document.createElement('p');
  el.textContent = 'Hello World';
  content.appendChild(el);
</script>
```

```
<div id=”new”>
  <p>
    Hello World
  </p>
</div>
```

JS: Creating Elements

```
<p>One</p>
<p>Two</p>
<p>Three</p>

<script>
  var paragraphs = document.body.getElementsByTagName("p");
  document.body.insertBefore(paragraphs[2], paragraphs[0]);
</script>
```

Result…

```
<p>Two</p>
<p>One</p>
<p>Three</p>
```

JS: attributes

attributes
Returns a live collection of all attribute nodes.

element.getAttribute(String name)
Returns the value of a specified attribute on the element.

element.hasAttribute(String name)
Returns a Boolean value indicating whether the specified element has the specified attribute or not.

```
<div class="myClass" id="myId" title="Some text title">Some text</div>

<script>
  var attrs = document.getElementById("myId").attributes;
</script>
```

```
<p data-classified="secret">The launch code is 00000000.</p>
<p data-classified="unclassified">I have two feet.</p>

<script>
  var paras = document.body.getElementsByTagName("p");
  var value = paras[0].getAttribute(‘data-classified’);
</script>
```

JS: Layout

Access to the layout: box, style,etc.

```
<p style="border: 3px solid red">I'm boxed in</p>

<script>
  var para = document.body.getElementsByTagName("p")[0];
  console.log("clientHeight:", para.clientHeight);
  console.log("offsetHeight:", para.offsetHeight);
</script>
```

```
<p id="para" style="color: purple">Pretty text</p>

<script>
  var para = document.getElementById("para");
  console.log(para.style.color);
  para.style.color = "magenta";
</script>
```

Handling events

→ Mouse events
    → click, wheel,contextmenu, mousedown, mouseup, ...
    → mouseenter, mouseover, mouseout, mousemove, ...
→ Keyboard events
    → keydown, keypress, keyup
→ Clipboard events
    → cut, copy, paste
→ Scroll events
...

Source: https://developer.mozilla.org/en-US/docs/Web/Events

Handling events

```
<button id="hello">Hello World</button>

<script type="text/javascript">
    document.addEventListener(
        "click",
        function(ev) {
            console.log("click");
        }
    );
    document.addEventListener("mouseover",function(e) {console.log("hover")});
    document.addEventListener("wheel",function(e) {console.log("wheel")});
</script>
```

Source: https://developer.mozilla.org/en-US/docs/Web/Events

Events and DOM nodes

```
<button>Hello Bordeaux</button>
<button id="hello">Hello World</button>

<script type="text/javascript">
  // Window level
  addEventListener("click",function(ev) {console.log("click in win");});

  // Document
  document.addEventListener("click",function(ev) {console.log("click doc");});

  // Element
  var el = document.getElementById('hello');
  el.addEventListener("click",function(ev) {console.log("click in el");});
</script>
```

JS: Handling events

Event objects

```
<button id="hello">Hello World</button>

<script type="text/javascript">
  // Element
  var el = document.getElementById(‘hello’);
  el.addEventListener(
    "mousedown",
    function(ev) {
      console.log(ev.which);
    }
  );
</script>

// Left button: 1, middle: 2, and right button: 3.
```

Event propagation

→ From the more specific toward the parent(s).
→ Outward propagation
→ stopPropagation() method in Event object