Desarrollo de Aplicaciones Web Empresariales

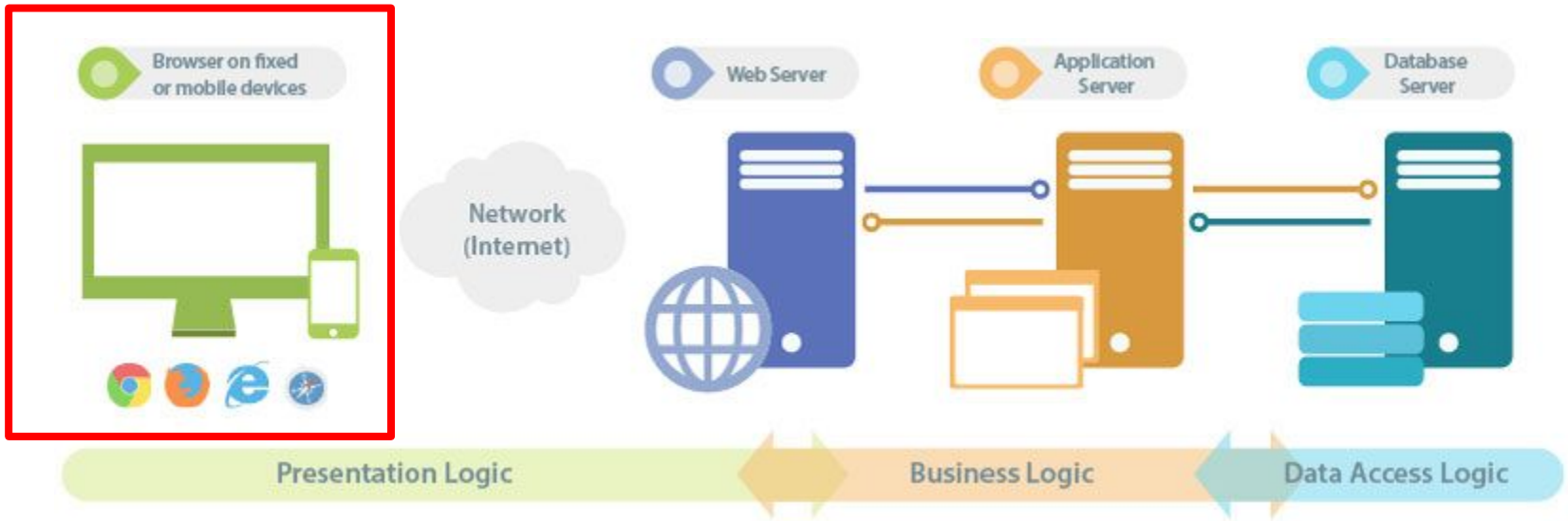
Exequiel Fuentes Lettura
exequiel.fuentes@ucn.cl

Información de contacto

- Exequiel Fuentes Lettura
  - Email: exequiel.fuentes@ucn.cl
  - Horario de Atención: Jueves y Viernes, bloque C

- Departamento de Ingeniería de Sistemas y Computación
  - Oficina: Y1 - 329
  - http://www.disc.ucn.cl

Arquitectura de aplicaciones Web: DOM



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

Document Object Model (DOM)

- Document Object Model (DOM) es una interfaz de programación para documentos HTML y XML.
- Proporciona una representación estructurada de un documento y define una manera que se pueda acceder a ella.
- Esta estructura es un grupo de nodos y objetos que tienen propiedades y métodos.
- Javascript puede consultar o modificar un documento HTML.

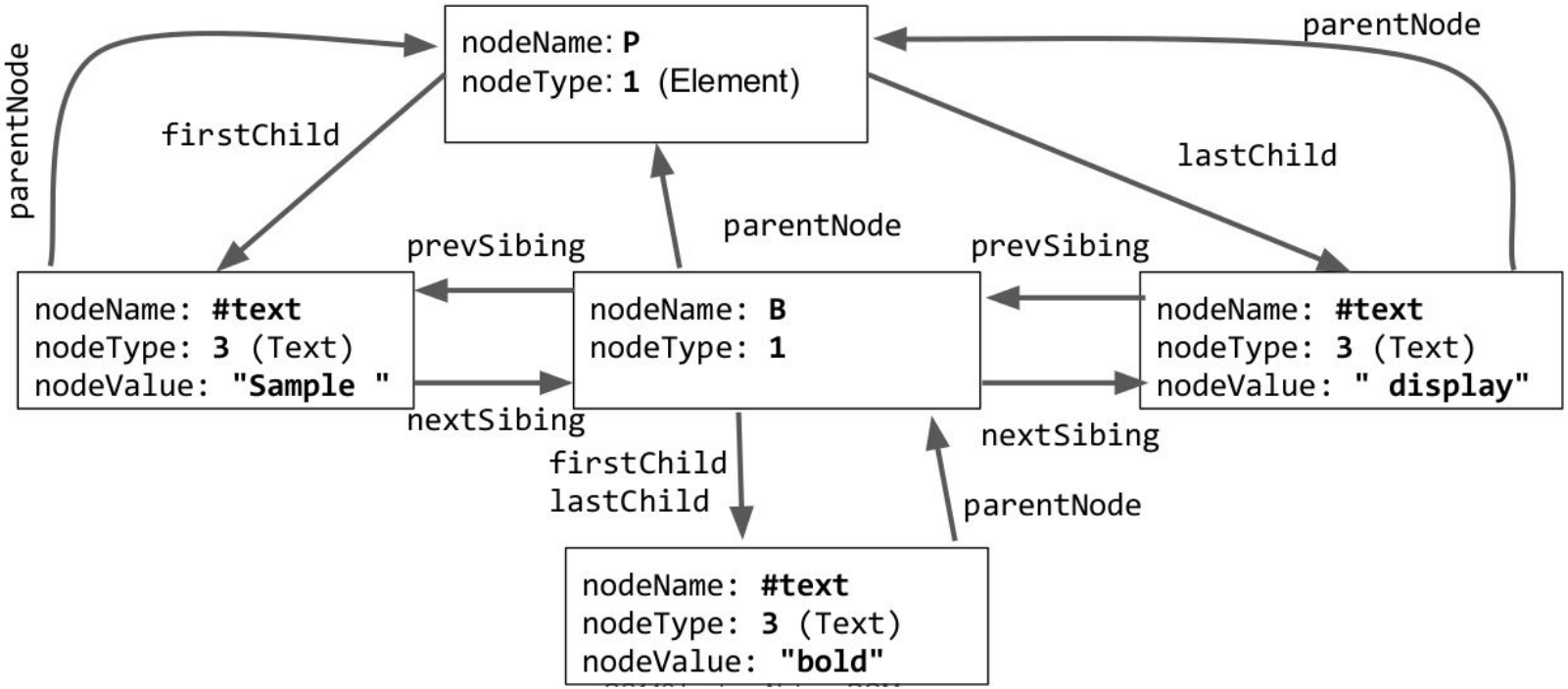
La jerarquía DOM

- Raíz: window.document
- Seguido por la estructura HTML:
  - window.document.head
  - window.document.body
- DOM tiene una gran cantidad de propiedades, la mayoría privadas.
- Los objetos (representación de elementos, textos, etc) tiene un conjunto común de propiedades y métodos llamado Nodo DOM.

DOM Node: métodos y propiedades

- Identificación:
  - Propiedad nodeName es el tipo de elemento (en mayúsculas: DIV, P, etc.) o #text
- Estructura jerárquica del documento
  - parentNode, nextSibling, prevSibling, firstChild, lastChild.
- Proporciona métodos para acceder y cambiar valores:
  - getAttribute, setAttribute, etc.

Sample bold display



Accediendo a un Node

- Se puede ir a través de la jerarquía, no se recomienda
  - element = document.body.firstChild.nextSibling.firstChild;
    element.setAttribute….
- Se debe utilizar métodos para encontrar elementos:
  - HTML: ...

    element = document.getElementById("div42");
    element.setAttribute….
- Hay otros métodos: getElementsByClassName(), getElementsByTagName(), etc.

Métodos y propiedades más comunes

- textContent: Texto del nodo y sus descendientes:
  - Ejemplo anterior: “Sample bold display”
- innerHTML: Describe los descendientes de un elemento en HTML:
  - Ejemplo anterior: “Sample bold display”
- outerHTML: Similar a innerHTML, pero incluye al elemento padre:
  - Ejemplo anterior: “ Sample bold display ”
- getAttribute()/setAttribute(): Obtiene o cambia un atributo de un elemento.

Operaciones para cambiar

- Cambiar el contenido de un elemento:
  - element.innerHTML = "This text is important";
    Reemplaza el contenido, pero mantiene los atributos.
- Cambiar un atributo, por ejemplo, de un elemento img
  - img.src="newImage.jpg";
- Hacer visible o invisible un elemento:
  - Invisible: element.style.display = "none";
  - Visible: element.style.display = "";

Interacciones del DOM con CSS

- Actualizar el atributo class:
  - element.className = "active";
- Actualizar el estilo de un elemento:
  - element.style.color = "#ff0000";
- Se puede obtener un elemento utilizando CSS:
  - document.querySelector() y document.querySelectorAll()
  - Ejemplo:
    - var el = document.querySelector(".myclass");
    - var el = document.querySelector("div.user-panel.main input[name=login]");

Cambiar la estructura del nodo

- Crear un nuevo elemento:
  - element = document.createElement("P");
  - element = document.createTextNode("My Text");
- Agregar un elemento a otro:
  - parent.appendChild(element);
  - parent.insertBefore(element, sibling);
- Remover un elemento:
  - node.removeChild(oldNode);
- Pero, usando innerHTML es más simple y eficiente.

Más operaciones

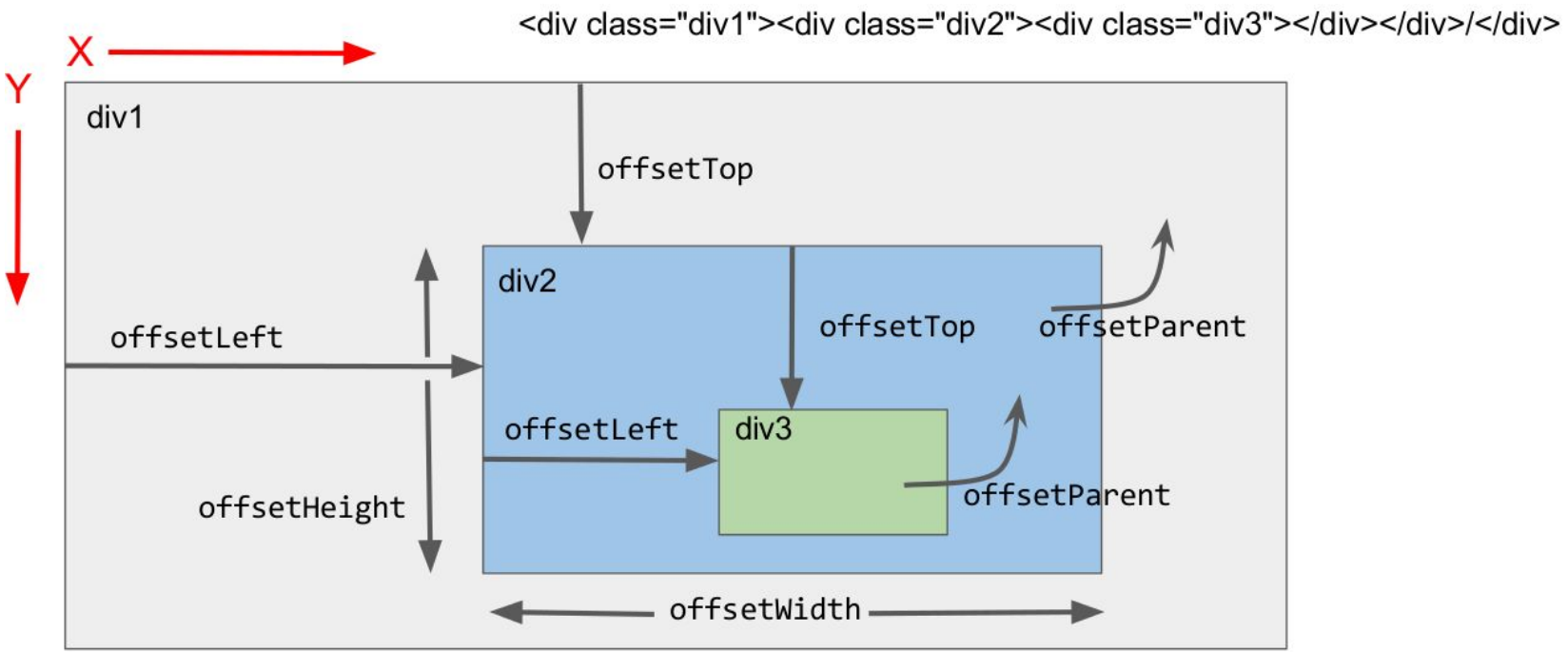
- Redirigir a una página nueva:
  - window.location.href = "newPage.html";

    Nota: Esto puede hacer que se pierda la actual ejecución del Javascript.
- Comunicación con el usuario:
  - console.log("Hola!!"); // Mensaje para el registro del browser.
  - alert(“Ha ocurrido un error!”);
  - confirm("Está seguro que quiere eliminar el usuario?");

Sistemas de coordenadas

- El origen de la pantalla es arriba a la izquierda.
- La posición de un elemento se determina por la esquina superior izquierda de su margen.
- Se obtiene la localización de un elemento usando:
  - element.offsetLeft, element.offsetTop
- Las coordenadas son relativas a element.offsetParent, la cual no necesariamente es el mismo que element.parentNode

Sistemas de coordenadas



Posicionando elementos

- Normalmente, los elementos son posicionados por el browser de acuerdo a la estructura del documento.
- Para poner un elemento fuera de su flujo dado por el documento, se puede hacer usando Javascript:
  - element.style.position = "absolute";

    element.style.left = "40px";

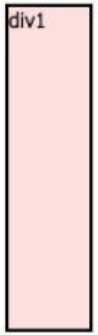
    element.style.top = "10px";
  - Usando "absolute" el elemento no ocupa espacio en el flujo del documento.
- El origen de un offsetParent (para posicionar sus descendientes) está justo en la esquina superior izquierda de su borde.

Dimensión de los elementos

- Para obtener la dimensión:
  - element.offsetWidth
  - element.offsetHeight
- Actualizar la dimensión:
  - element.style.width
  - element.style.height

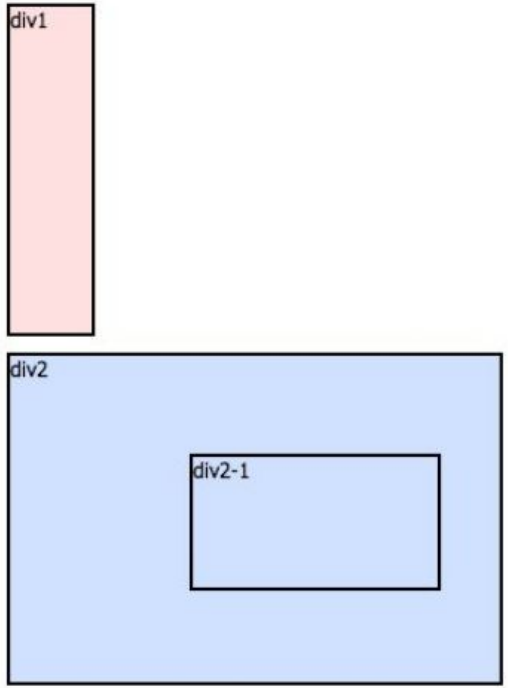
Posicionando

```
<body>
    <div id="div1">
        <p>div1</p>
    </div>

#div1 {
   width: 50px;
   height: 200px;
   background: #ffe0e0;
}
```



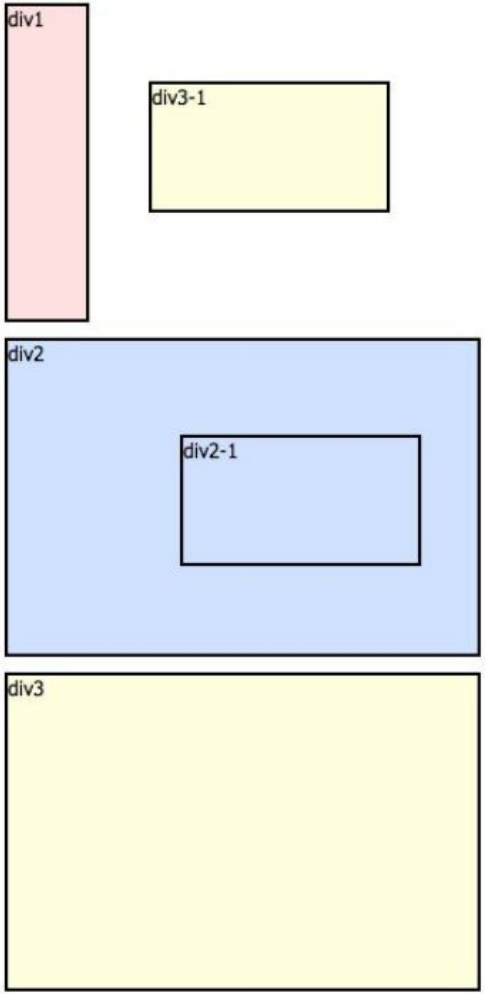
Posicionando

```
    ...
    <div id="div2">
       <p>div2</p>
       <div id="div2-1">
          <p>div2-1</p>
       </div>
    </div>
#div2 {width: 300px; height:
200px; position: relative;
    background: #d0e0ff;}

#div2-1 {width: 150px; height:
80px; position: absolute;
   top: 50px; left: 100px;
background: #d0e0ff;}
```



Posicionando

```
...
    <div id="div3">
       <p>div3</p>
       <div id="div3-1">
         <p>div3-1</p>
       </div>
    </div>
#div3 {width: 300px; height:
200px; background: #ffffe0;}

#div3-1 {width: 150px; height:
80px; position: absolute; top:
50px; left: 100px; background:
#ffffe0;}
```



Posicionando

```
<html>
  <head>
    <meta charset="utf-8">
    <title>Testing positioning context</title>
    <style>
      div {
        border: 2px solid black;
      }
      #div1 {
        width: 50px;
        height: 200px;
        background: #ffe0e0;
      }
      #div2 {
        width: 300px;
        height: 200px;
        position: relative;
        background: #d0e0ff;
      }
      #div2-1 {
        width: 150px;
        height: 80px;
        position: absolute;
        top: 50px;
        left: 100px;
        background: #d0e0ff;
      }
      #div3 {
        width: 300px;
        height: 200px;
        background: #ffffe0;
      }
      #div3-1 {
        width: 150px;
        height: 80px;
        position: absolute;
        top: 50px;
        left: 100px;
        background: #ffffe0;
      }
    </style>
  </head>
  <body>
    <div id="div1">
      <p>div1</p>
    </div>
    <div id="div2">
      <p>div2</p>
      <div id="div2-1">
        <p>div2-1</p>
      </div>
    </div>
    <div id="div3">
      <p>div3</p>
      <div id="div3-1">
        <p>div3-1</p>
      </div>
    </div>
  </body>
</html>
```

¿Preguntas?