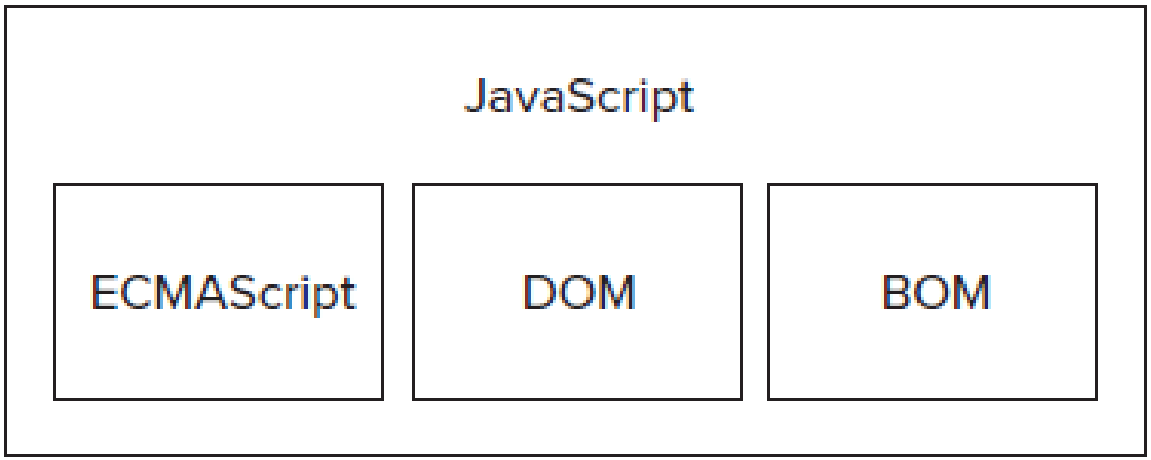
CFGS DAW

Mòdul 6: Desenvolupament web en entorn client

INS Joan d'Àustria

Javascript



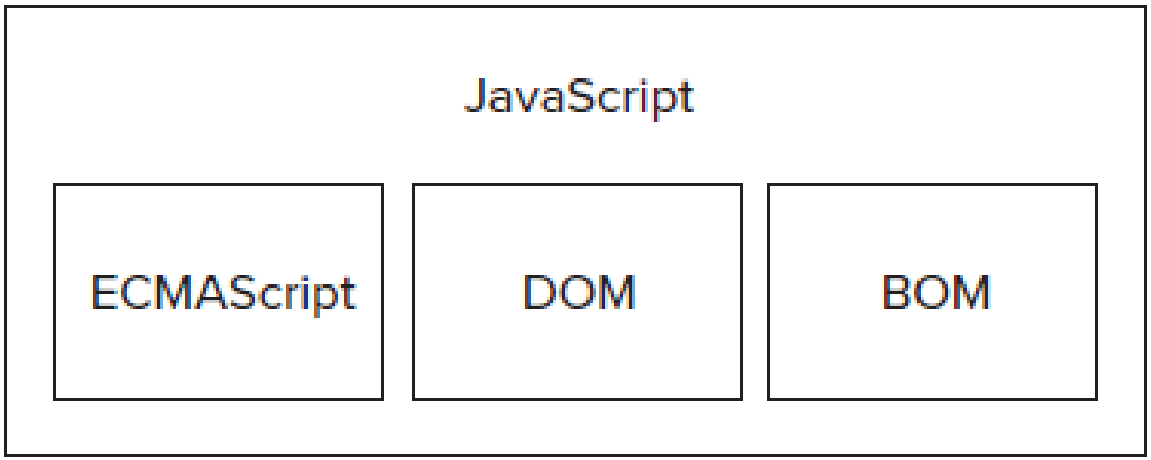
- ECMAScript: Ecma262 es una especificación de un lenguaje. Javascript, Jscript (Microsoft) o Action Script (Adobe) no son más que implementaciones de este lenguaje y los navegadores no son más que un huesped en el que esta implementación puede existir

Javascript

BROWSER	ECMASCRIPT COMPLIANCE
Netscape Navigator 2	—
Netscape Navigator 3	—
Netscape Navigator 4–4.05	—
Netscape Navigator 4.06–4.79	Edition 1
Netscape 6+ (Mozilla 0.6.0+)	Edition 3
Internet Explorer 3	—
Internet Explorer 4	—
Internet Explorer 5	Edition 1
Internet Explorer 5.5–7	Edition 3
Internet Explorer 8	Edition 5*
Internet Explorer 9+	Edition 5
Opera 6–7.1	Edition 2
Opera 7.2+	Edition 3
Safari 1–2.0.x	Edition 3*
Safari 3.x	Edition 3
Safari 4.x–5.x	Edition 5*
Chrome 1+	Edition 3
Firefox 1–2	Edition 3
Firefox 3.0.x	Edition 3
Firefox 3.5–3.6	Edition 5*
Firefox 4+	Edition 5

Soporte ECMA de los diferentes navegadores

Javascript



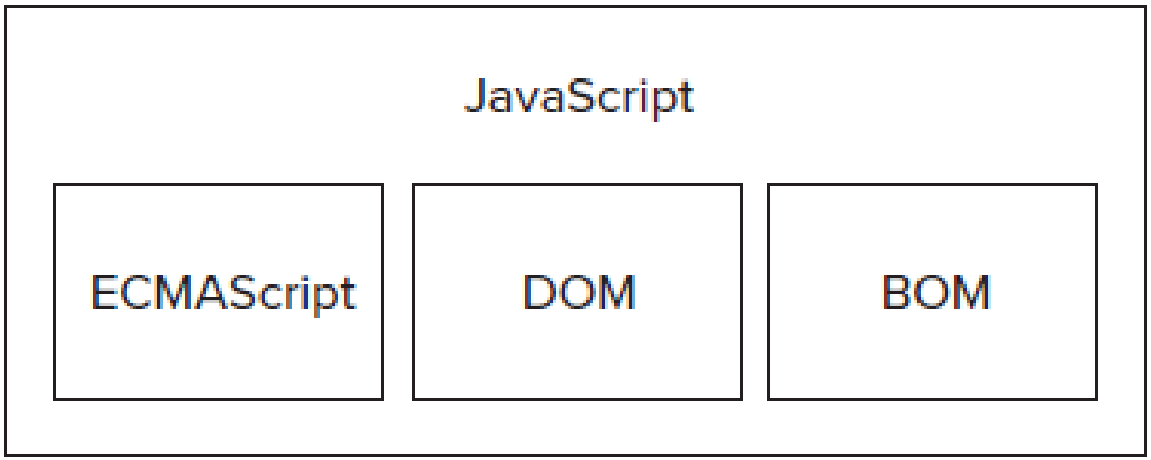
- DOM: Estructura jerárquica del documento.
  - Nivel 1: Mapea la estructura del documento
  - Nivel 2: Añade soporte a eventos de usuario y CSS
  - Nivel 3: Añade métodos de validación y soporte XML

Javascript

BROWSER	DOM COMPLIANCE
Netscape Navigator 1.–4.x	—
Netscape 6+ (Mozilla 0.6.0+)	Level 1, Level 2 (almost all), Level 3 (partial)
Internet Explorer 2–4.x	—
Internet Explorer 5	Level 1 (minimal)
Internet Explorer 5.5–8	Level 1 (almost all)
Internet Explorer 9+	Level 1, Level 2, Level 3
Opera 1–6	—
Opera 7–8.x	Level 1 (almost all), Level 2 (partial)
Opera 9–9.9	Level 1, Level 2 (almost all), Level 3 (partial)
Opera 10+	Level 1, Level 2, Level 3 (partial)
Safari 1.0.x	Level 1
Safari 2+	Level 1, Level 2 (partial)
Chrome 1+	Level 1, Level 2 (partial)
Firefox 1+	Level 1, Level 2 (almost all), Level 3 (partial)

Soporte DOM de los diferentes navegadores

Javascript



- BOM: Permite tratar y manipular la ventana del navegador
  - No existe ningun estandard para tratarlo hasta la aparición de HTML5

Javascript y HTML

- El elemento <script>:
  - async (opcional): Indica que el script debe empezar a descargarse immediatamente pero no bloquea al navegador, permitiendo que se continue descargando otros elementos. Sólo es válido para scripts externos
  - defer (opcional): Indica que la ejecución del script se puede postergar hasta que el documento haya sido cargado completamente
  - src (opcional): Indica que el código está almacenado en un archivo externo y se indica su ruta.
  - type (opcional): Indica el tipo de contenido (MIME) usado en el script. Si se omite se asume “text/javascript”.

Javascript y HTML

- Igual que en CSS, podemos incluir código directamente o llamar (src) a un archivo externo

```
<script type="text/javascript">
alert("Hola mundo");
</script>
```

```
<script type="text/javascript" src="/js/ejemplo.js"></script>
```

¿Qué pasa si los combinamos?

```
<script type="text/javascript" src="/js/ejemplo.js">
alert("Hola mundo");
</script>
```

Javascript y HTML

- El código se descarga e interpreta línea a línea
- Tradicionalmente, todos los scripts y CSS se sitúan en el <head>:
  - Mantener referencias externas HTML en un mismo lugar
  - Descargar e interpretar todo el código javascript antes del renderizado de la página
- Si una página requiere mucho javascript, esto puede provocar lentitud, pues hasta que no se compile todo, no se empieza a cargar la página:
  - Incluir algunas llamadas como primera línea después de <body>
  - Utilizar el atributo defer (la ejecución comenzará después de haber renderizado </html>)
  - Utilizar async (si hay más de un script no se garantiza el orden. No se debe utilizar para scripts que modifiquen el DOM)

Javascript y HTML

- ¿Dentro o fuera de la página?
  - Se recomienda en general el uso de javascript como elemento externo. Pese a que no hay aumento de rendimiento, los motivos son:
    - Mantenibilidad: Mantener código repartido por todos los documentos HTML es muy costoso
    - Portabilidad: Muchas funciones pueden ser compartidas por varias páginas
    - Cooperación: Tener todo el código en un directorio /js facilita el trabajo en equipo
    - Caché: Si dos páginas necesitan el mismo archivo, este sólo se necesitará cargar una vez, por lo que aumentará la velocidad de carga
    - Facilidad: Javascript en documentos externos no necesita diferenciar si el documento es HTML o XHTML

Javascript y HTML

- Javascript interno en documentos XHTML:
  - XHTML es muy estricto:

```
…
If (x < y ) {
…
```

El símbolo < significa que se abre etiqueta HTML, por lo que genera un error. Soluciones:

```
…
if (x &lt; y) {
…
```

```
<script type="text/javascript"><![CDATA[
…
if (x < y) {
…
]]></script>
```

Javascript y HTML

- El elemento <noscript>:

Especialmente para navegadores antiguos que no soportaban Javascript o para los actuales que los usuarios tengan deshabilitada la ejecución, se creó la etiqueta <noscript>

```
<body>
<noscript>
<p>La página no soporta javascript.</p>
</noscript>
</body>
```