



JVN Web

Integrantes:

Wilson Nicolas Arevalo, Víctor Alfredo Barragán, Jonathan López Castellanos .

Propósito:

JVN Web es un lenguaje creado para facilitar los primeros pasos en el desarrollo de aplicaciones o páginas web, en él se condensan 3 herramientas populares a la hora de crear una página web simple o sencilla (HTML, CSS , Javascript) con lo cual se facilitan las primeras inmersiones en esta rama del desarrollo web.

Traducción:

La sintaxis de JVN Web fue desarrollada pensando en la simplicidad, basándose en la sintaxis de html, css y javascript e inspirados por el proyecto del lenguaje Psicoder.
La traducción se realiza mediante visitors utilizando la herramienta ANTLR

Estructura:

Los elementos de JVNWeb tienen la sintaxis:

NombreDelElemento
(clases: “lista_clases”,
estilo: nombreDelEstiloX es valor y nombreDelEstiloY es valor ... ,
nombreDelEventoX: nombreDeLaFunción ...)

ElementosAnidados (Si es un elemento contenedor)
FinNombreDelElemento

Nota: Algunos estilos no tienen ningún valor, sino son banderas para indicar que se aplica el estilo (es decir, solo se especifica el nombre del estilo)

Los archivos tienen la siguiente estructura:

Encabezado
Título

<p>Título de la Página</p> <p><i>FinTitulo</i></p> <p><i>FinEncabezado</i></p> <p><i>Cuerpo</i></p> <p><i>NombreDelElemento</i></p> <p>(clases: “lista_clases”, otrosAtributos: “valor”, estilo: nombreDelEstiloX es valor y nombreDelEstiloY es valor ... , nombreDelEventoX: nombreDeLaFunción ...</p> <p>ElementosAnidados (Si es un elemento contenedor)</p> <p><i>FinNombreDelElemento</i></p> <p>...</p> <p><i>FinCuerpo</i></p>
<p>Nota: Algunos estilos no tienen ningún valor, sino son banderas para indicar que se aplica el estilo (es decir, solo se especifica el nombre del estilo)</p>

Elementos:

Los elementos de JVNWeb se dividen entre contenedores y elementos simples:

Contenedor
<p><i>Contenedor</i></p> <p>Contenido</p> <p><i>FinContenedor</i></p>
<p><u>Descripción:</u></p> <p>Se traduce a <div>Contenido</div></p> <p><u>Eventos:</u></p> <ul style="list-style-type: none"> - alHacerClic - alApuntar - alSalir

Formulario
<p><i>Formulario</i></p> <p><i>(alEnviarDatos: nombreDeLaFunción)</i></p> <p>Contenido</p> <p><i>FinFormulario</i></p>

Descripción:

Se traduce a `<form onsubmit="nombreDeLaFunción">Contenido</form>`

Eventos:

- alEnviarDatos (maneja la acción a tomar cuando se envía los datos del formulario)
- alHacerClic
- alApuntar
- alSalir

Párrafo

Parrafo

Texto

Texto

Texto

FinTexto

...

FinParrafo

Descripción:

Se traduce a `<p>TextoTexto...</p>`

Eventos:

- alHacerClic
- alApuntar
- alSalir

Lista Ordenada

ListaOrdenada

ElementoLista

...

ElementoLista

FinListaOrdenada

Descripción:

Se traduce a ``

Eventos:

- alHacerClic

<ul style="list-style-type: none"> - alApuntar - alSalir
--

<i>Lista Sin Orden</i>
<i>ListaSinOrden</i> <i>ElementoLista</i> ... <i>ElementoLista</i> <i>FinListaSinOrden</i>
<u>Descripción:</u> Se traduce a <u>Eventos:</u> <ul style="list-style-type: none"> - alHacerClic - alApuntar - alSalir

<i>Elemento de Lista</i>
<i>ElementoLista</i> Contenido <i>FinElementoLista</i>
<u>Descripción:</u> Se traduce a <u>Eventos:</u> <ul style="list-style-type: none"> - alHacerClic - alApuntar - alSalir

<i>Tabla</i>
<i>Tabla</i> <i>EncabezadoTabla</i> <i>FilaTabla</i> ...

<i>FilaTabla</i> <i>FinTabla</i>
<u>Descripción:</u> Se traduce a <table></table>
<u>Eventos:</u> <ul style="list-style-type: none"> - alHacerClic - alApuntar - alSalir

<i>Encabezado de Tabla</i>
<i>EncabezadoTabla</i> <i>FilaTabla</i> <i>ElementoTabla</i> <i>FilaTabla</i> ... <i>ElementoTabla</i> <i>FinEncabezadoTabla</i>
<u>Descripción:</u> Se traduce a <th></th>
<u>Eventos:</u> <ul style="list-style-type: none"> - alHacerClic - alApuntar - alSalir

<i>Fila de Tabla</i>
<i>FilaTabla</i> <i>ElementoTabla</i> ... <i>ElementoTabla</i> <i>FinFilaTabla</i>
<u>Descripción:</u> Se traduce a <tr></tr>
<u>Eventos:</u>

<ul style="list-style-type: none"> - alHacerClic - alApuntar - alSalir

<i>Elemento de Tabla</i>
<i>ElementoTabla</i> Contenido <i>FinElementoTabla</i>
<u>Descripción:</u> Se traduce a <td></td>
<u>Eventos:</u> <ul style="list-style-type: none"> - alHacerClic - alApuntar - alSalir

<i>Enlace</i>
<i>Enlace</i> (referencia: “localizadorDelRecurso”) Texto <i>FinEnlace</i>
<u>Descripción:</u> Se traduce a
<u>Eventos:</u> <ul style="list-style-type: none"> - alHacerClic - alApuntar - alSalir

<i>Línea</i>
<i>Linea</i> <i>FinLinea</i>
<u>Descripción:</u>

Se traduce a **<hr>**

Eventos:

- alHacerClic
- alApuntar
- alSalir

Texto con Enfasis

TextoConEnfasis ***enfasis***

Texto

FinTextoConEnfasis

Descripción:

Se traduce a **<h#></h#>** según el énfasis especificado. El ***enfasis*** es el número correspondiente (entre 1 y 6)

Eventos:

- alHacerClic
- alApuntar
- alSalir

Botón

Boton

Texto

Texto

FinBoton

Descripción:

Se traduce a **<button></button>**

Eventos:

- alHacerClic
- alApuntar
- alSalir

Etiqueta

<i>Etiqueta</i> Texto <i>Texto</i> <i>FinEtiqueta</i>
<u>Descripción:</u> Se traduce a <label></label>
<u>Eventos:</u> <ul style="list-style-type: none"> - alHacerClic - alApuntar - alSalir

<i>Entrada de Texto</i>
<i>EntradaDeTexto</i> (nombre: “nombreDeLaEntrada”, valor: “valorDeLaEntrada”, descripcion: “placeholder”, alModificar: nombreDeLaFunción, alSeleccionar: nombreDeLaFunción) <i>FinEntradaDeTexto</i>
<u>Descripción:</u> Se traduce a <input name=”” value=”” placeholder=””>
<u>Eventos:</u> <ul style="list-style-type: none"> - alModificar - alSeleccionar - alHacerClic - alApuntar - alSalir

<i>Area de Texto</i>
<i>AreaDeTexto</i> (nombre: “nombreDeLaEntrada”, alModificar: nombreDeLaFunción,

<i>alSeleccionar: nombreDeLaFunción)</i> Texto <i>Texto</i> <i>FinAreaDeTexto</i>
<u>Descripción:</u> Se traduce a <code><textarea name=""></code> <u>Eventos:</u> <ul style="list-style-type: none"> - alModificar - alSeleccionar - alHacerClic - alApuntar - alSalir

Selector
<i>Selector</i> <i>(nombre: "nombreDelSelector",</i> <i>valor: "valorDelSelector",</i> <i>evento: alModificar: nombreDeLaFunción,</i> <i>evento: alSeleccionar: nombreDeLaFunción)</i> <i>Opcion</i> <i>FinSelector</i>
<u>Descripción:</u> Se traduce a <code><select name="" value="" placeholder=""></select></code> <u>Eventos:</u> <ul style="list-style-type: none"> - alModificar - alSeleccionar - alHacerClic - alApuntar - alSalir

Opción
<i>Opcion</i> <i>(valor: "valorDeLaEntrada")</i> Texto

<p><i>Texto</i></p> <p><i>FinOpcion</i></p>
<p><u>Descripción:</u></p> <p>Se traduce a <option value=""></option></p> <p><u>Eventos:</u></p> <ul style="list-style-type: none"> - alHacerClic - alApuntar - alSalir

<p><i>Audio</i></p>
<p><i>Audio</i></p> <p><i>(conControles)</i></p> <p><i>Fuente</i></p> <p><i>FinAudio</i></p>
<p><u>Descripción:</u></p> <p>Se traduce a <audio controls></audio></p> <p><u>Eventos:</u></p> <ul style="list-style-type: none"> - alHacerClic - alApuntar - alSalir

<p><i>Video</i></p>
<p><i>Video</i></p> <p><i>(conControles)</i></p> <p><i>Fuente</i></p> <p><i>FinVideo</i></p>
<p><u>Descripción:</u></p> <p>Se traduce a <video controls></video></p> <p><u>Eventos:</u></p> <ul style="list-style-type: none"> - alHacerClic - alApuntar - alSalir

<i>Fuente</i>
<i>Fuente</i> (fuente: "recurso", tipo: "tipo") <i>FinFuente</i>
<u>Descripción:</u> Se traduce a <source src="" type="">
<u>Eventos:</u> <ul style="list-style-type: none"> - alHacerClic - alApuntar - alSalir

<i>Imagen</i>
<i>Imagen</i> (fuente: "recurso", textoAlternativo: "texto", ancho: #medida en pixeles o %, alto: #medida en pixeles o %") <i>FinImagen</i>
<u>Descripción:</u> Se traduce a
<u>Eventos:</u> <ul style="list-style-type: none"> - alHacerClic - alApuntar - alSalir

<i>Texto</i>
<i>Texto</i> Texto <i>Texto</i> <i>Texto</i>
<u>Descripción:</u>

Se traduce a ``

Eventos:

- alHacerClic
- alApuntar
- alSalir

Estilos

Los estilos son traducidos directamente a css, por lo tanto los estilos introducidos son un conjunto de los estilos permitidos en este lenguaje.

Los estilos introducidos en el lenguaje con su sintaxis y traducción son:


Css	JVN Web
width	ancho
height	alto
min-width	anchoMinimo
max-width	anchoMaximo
min-height	alturaMinima
max-height	alturaMaxima
position	posicion
position	ubicacion
display	visualizacion
z-index	profundidad
float	flotamiento
aling	alineado
justify	justificado
border	borde
cursor	cursor
padding	espaciado
color	color
background-color	colorFondo

opacity	opacidad
size	tamano
family	familia

Los colores se establecen algunos colores predeterminados como :

Color	RGB hexadecimal	muestra
rojo	#ff0000	
verde	#008000	
azul	#0000ff	
amarillo	#ffff00	
violeta	#ff00ff	
negro	#000000	
marron	#a52a2a	
gris	#808080	
naranja	#ffa500	
rosa	#ff1493	
purpura	#800080	
blanco	#ffffff	

Adicionalmente se agrega la posibilidad de trabajar con RGB para escoger el color de preferencia mediante la sintaxis:

(estilo: color es # rojo 4 verde 132 azul 250)	
---	---

Ejemplo de estilos:

Establecer los estilos básicos de dos títulos diferentes

JVN Web
<i>Encabezado</i> <i>Título</i> <i>'Título de la página'</i>

FinTitulo
FinEncabezado
Cuerpo
TextoConEnfasis 1
(estilos: colorFondo es amarillo y color es azul y alto es 50 pixeles)
FinTextoConEnfasis
Parrafo
(estilos: colorFondo es rojo y ancho es 100 pixeles y alineado es centro)
FinParrafo
FinCuerpo

Eventos:

Los eventos configurados en JVN Web son los más usados o los más comunes en Javascript cuando se busca crear una página web , estos son:

Javascript	JVN Web
onclick	alHacerClic
onmouseover	alApuntar
onmouseout	alSalir
onchange	alModificar
onselect	alSeleccionar
onsubmit	alEnviarDatos

La creación de los eventos se hace mediante la siguiente sintaxis:

(alHacerClic : id)

Nota: El id es la referencia a una función previamente creada

Traducción:

HTML

```

<!DOCTYPE html>
<html lang="es">
<head>
<title>
Titulo de la pagina

```

```

</title>
<meta charset="UTF-8">
</head>
<body>
<h1 style="background-color: #ffff00; color: #0000ff; height: 50px ; " >
</h1>
<p style="background-color: #ff0000; width: 100px ; align-items: center; " >
</p>
</body>
</html>

```

Estructura de código JS:

La gramática del código referente a los script de JavaScript en JVNWeb tiene sus bases en la desarrollada para el lenguaje PsiCoder, cuyo manual de referencia se puede encontrar en el siguiente documento: [Lenguaje PsiCoder](#).

Se modificaron y agregaron algunos elementos a esta, teniendo en cuenta que PsiCoder no permite abarcar algunos de los elementos bases de JavaScript.

Es necesario agregar la sección de *Codigo* a la estructura del archivo si se desea agregar un script de JVNWeb.

```

Encabezado
  Titulo
    Título de la Página
  FinTitulo
FinEncabezado
Cuerpo
...
FinCuerpo
Codigo
...
FinCodigo

```

Algunos de los elementos agregados a JVNWeb que hacen parte de JavaScript son:

Arreglo

Los arreglos pueden tener uno o más elementos de diferentes tipos de datos.

Estructura: [*valor₁*, *valor₂*, ... , *valor_n*]

Declaración: Se usa la palabra reservada “lista” y el identificador del arreglo:

lista *identificador* = [1,2,3,4]

Acceso: Se puede acceder a un elemento del arreglo a partir de su posición:
identificador[*posición*] La posición puede ser un número entero o una variable.

Objeto

Los objetos poseen propiedades que pueden ser de diferentes tipos de datos.

Estructura: objeto *identificador* tiene *propiedades*

Declaración: Se usan la palabras reservadas “objeto, tiene, fin_objeto”, el identificador de objeto (de ser necesario) y sus propiedades:

objeto *identificador* tiene *propiedad₁ : valor* , *propiedad₂ : valor* ... fin_objeto

En caso de que el objeto tenga más de una propiedad, estas deben ir separadas por comas.

Si un objeto tiene otro objeto como una de sus propiedades, es necesario añadir el carácter ‘:’ antes de declarar la propiedad:

objeto *casa* tiene *entrada : `azul`* , **:objeto** cuarto tiene puertas : 2 **fin_objeto** , garaje : `grande` fin_objeto

Acceso: Se puede acceder a las propiedades del objeto a partir del identificador de la propiedad:

identificadorObjeto.nombrePropiedad : *casa.entrada*

identificadorObjeto[“*nombrePropiedad*”] : *casa*[“*entrada*”]

Si se tiene un objeto dentro de un arreglo, se omite el identificador de objeto.

lista arregloObjetos : [*objeto tiene propiedad: valor finObjeto* , ...]

Declaración de variables

variable identificador = expresion

o

variable identificador₁ = expresion, identificador₂ = expresion, ...

Descripción:

Se traduce a **var identificador = expresión; o**

var identificador₁ = expresión, identificador₂ = expresión, ... ;

Obtención de elemento o clase

asignarElemento (elemento o clase) a identificador

Descripción:

Se traduce a **var identificador = document.querySelector(elemento o clase);**

Los elementos pueden ser: ‘Contenedor’, ‘Formulario’, ‘Parrafo’, etc.
La clase debe empezar por ‘.’: `.clase`

Cambio de HTML interno de un elemento

(identificador o elementoArreglo) nuevoContenido : valor

Descripción:

Se traduce a **(identificador o elementoArreglo).innerHTML = valor;**

Donde elementoArreglo : arreglo[posición]

Cambio de estilo de un elemento

*(identificador o elementoArreglo) nuevoEstilo : (nombreEstilo ‘es’ valorEstilo) o
estiloBooleano*

Descripción:

Se traduce a **(identificador o elementoArreglo).style.nombreEstilo = valorEstilo;**

Donde elementoArreglo : arreglo[posición]

Ciclos para en

*para (llave en objeto) hacer
 contenido
 (romper o continuar;) (opcional)
fin_para*

Descripción: Permite iterar sobre objetos y arreglos.

Se traduce a **for (let llave in objeto) {
 contenido
 (break/continue;) (opcional)
}**

Ciclos para de

*para (variable de iterable) hacer
 contenido
 (romper o continuar;) (opcional)*

<i>fin_para</i>
<u>Descripción:</u> Permite iterar sobre arreglos y cadenas. Se traduce a for (let variable of iterable) { contenido (break/continue;) (opcional) }

<i>Impresión por consola</i>
<i>imprimir expresión fin_imprimir</i>
<u>Descripción:</u> Se traduce a console.log(expresión);

<i>Funciones</i>
<i>funcion (identificadorOpcional) (argumentos) hace</i> <i>contenido</i> <i>(retornar expresion;) (opcional)</i> <i>fin_funcion</i>
<u>Descripción:</u> Se traduce a function identificador (argumentos) { contenido return expresión; (opcional) }