


ESTANDARES DE DESARROLLO EN JAVASCRIPT


Versión 1.0

T.I.C.

| | | | |
|-----------------------------------------------------------------------------------|-------------------------------------------------|--|------------------|
|  | Estándares de Desarrollo para JavaScript | | |
| | Dirección: Tecnología de la Información | | |
| | Versión: 1.0 | | Mayo 2019 |


REVISIONES

| Fecha | Versión | Descripción | Autor |
|-------------|---------|------------------|--------------|
| 6/Mayo/2019 | 1.0 | Versión original | Samuel Arone |
| | | | |
| | | | |
| | | | |
| | | | |
| | | | |
| | | | |
| | | | |
| | | | |
| | | | |
| | | | |

| | | |
|-----------------------------------------------------------------------------------|-------------------------------------------------|------------------|
|  | Estándares de Desarrollo para JavaScript | |
| | Dirección: Tecnología de la Información | |
| | Versión: 1.0 | Mayo 2019 |

INDICE

| | |
|-----------------------------------------------------------------|-----------|
| OBJETIVOS | 4 |
| 1. ITEMS CONSIDERADOS PARA ESTANDARIZACION EN JAVASCRIPT | 4 |
| 1.1 Convenciones y Estándares de Nombres | 4 |
| 1.2 Definición de tipos de en las variables | 5 |
| 1.3 Uso de variables y funciones de ECMA 6 | 6 |
| 2.0 FUNCIONES UTILES | 6 |
| 2.1 Llamadas con AJAX - Get | 6 |
| 2.2 Llamadas con AJAX - Post | 7 |
| 2.3 Llamadas con AJAX _Go_Url | 7 |
| 3.0 HELPER – VALIDADORES | 8 |
| 3.1 _rules | 8 |
| 3.2 _required | 9 |
| 3.3 Empty | 10 |
| 3.4 Parse | 10 |
| 3.5 CSVtoJSON | 11 |
| 3.6 _comboFromCSV | 11 |
| 3.7 _comboFromJSON | 11 |
| 3.8 _comboltem | 11 |
| 3.9 _convertDate_ANSI | 12 |
| 3.10 _getParameter | 12 |
| 3.11 _par | 12 |
| 3.12 _modalBody | 13 |
| 3.13 swal | 13 |
| 4.0 ESTRUCTURA DEL ARCHIVO JAVASCRIPT | 13 |

| | | |
|-----------------------------------------------------------------------------------|-------------------------------------------------|------------------|
|  | Estándares de Desarrollo para JavaScript | |
| | Dirección: Tecnología de la Información | |
| | Versión: 1.0 | Mayo 2019 |

ESTANDARES EN JAVASCRIPT

OBJETIVOS

Implementar el uso de estándares con las mejores prácticas establecidas por los desarrolladores de WTS.

1. ITEMS CONSIDERADOS PARA ESTANDARIZACION EN JAVASCRIPT


1.1 Convenciones y Estándares de Nombres

| | |
|--------------------|-------------------------------------------------------------|
| Comentarios | Listado convenciones para todos los tipos de Objetos en C#. |
| Razón | Para facilitar la lectura de código en JavaScript. |

| Object Name | Notation | Plural | Abbreviation |
|------------------|-------------------|-----------|--------------|
| Class name | PascalCase | No | No |
| Method name | camelCase | Yes | No |
| Method arguments | camelCase | Yes | Yes |
| Local variables | camelCase | Yes | Yes |
| Constants name | camelCase | Yes | Yes |
| Field name | camelCase | Yes | Yes |
| Properties name | camelCase | Yes | Yes |
| | | | |

| | |
|--------------------|---------------------------------------------------------------------------------------------------------|
| | Notación PascalCase |
| Comentarios | El primer carácter de todas las palabras se escribe en Mayúsculas y los otros caracteres en minúsculas. |
| | |

| | |
|--------------------|-----------------------------------------------------------------------------------------------------------------------|
| | Notación camelCase |
| Comentarios | El primer carácter de todas las palabras a excepción de la primera se escribe en mayúscula y las demás en minúsculas. |
| | |

| | | |
|-----------------------------------------------------------------------------------|------------------------------------------|-----------|
|  | Estándares de Desarrollo para JavaScript | |
| | Dirección: Tecnología de la Información | |
| | Versión: 1.0 | Mayo 2019 |

```
// clase [PascalCase]
class Carro {
  // parametros [camelCase]
  constructor(marca, modelo, color) {

    //propiedades [camelCase]
    this.marca = marca;
    this.modelo = modelo;
    this.color = color;
  }

  // metodo [camelCase]
  getNombre() {
    return this.marca + " " + this.modelo;
  }
}

// variable [camelCase]
const carro = new Carro("Honda", "Accord", "Blue");
```

1.2 Definición de tipos de en las variables

| | |
|-------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Comentarios | El uso de variables globales solo se debe usar una por página. |
| Razón | Las variables globales de página se deben agrupar en una sola variable como objeto literal, esto ayuda a ordenar y acceder rápidamente a todas sus propiedades en un solo repositorio. |

Ejemplo

```
// Evitar
var codigo;
var correo;
var esHabilitado;

// Correcto
var oVariables={
  codigo:'',
  correo:'',
  esHabilitado:''
}
```

1.3 Uso de variables y funciones de ECMA 6

| variable / metodo | Comentario |
|---------------------------|------------------------------------------------------------------------------------------------------|
| 1. let: | Declarar las variables let dentro de una función. |
| 2. var: | Declarar las variables globales en la parte superior de la página como objeto literal. |
| 3. const: | Declarar variable de tipo constante para no alterar la información de la variable. |
| 4. arrow function: | Usar funciones de tipo flecha nos ayudará a tener una sintaxis limpia y ahorra líneas de código. |
| 5. Array: | El uso de Array de Objetos facilita la utilización de todos los métodos nativos del Array en ECMA 6. |
| 5.1. Map | Recorre y devuelve un nuevo array |
| 5.2. forEach | Recorrer un array por su objeto |
| 5.3. for | Recorrer un array por su índice |
| 5.4. some | Búsqueda si existe una coincidencia dentro de un array. |
| 5.5. reduce | Suma los valores de un array |
| 5.6. sort | Ordenar un array de forma ascendente o descendente |
| 5.7. find | Búsqueda de un objeto en un array por condición |
| 5.8. filter | Filtrar un array por condición |
| 5.9. Set: | Usar set para obtener una colección de valores únicos, es decir valores que no se repitan. |

2.0 FUNCIONES UTILES

2.1 Llamadas con AJAX - Get

| | |
|-------------|---------------------------------------------------------------------------------------------------------------------------------------------------|
| Comentarios | Carga de contenido, data o archivos con el método "Get". |
| Razón | <ul style="list-style-type: none"> Comunicación del Cliente con el servidor Traer contenido con poca o mucha información. |

```
// Ejemplo 1: Sin parametros
const url='Facturas/FacturaCliente/Index';
Get(url, fn_CargaFacturaCliente);

// Ejemplo 2: Con parametros
const parametroJSON = {idFactura:17,idCliente:7};
const url = 'Factura/FacturaCliente/Index?par='+ JSON.stringify(parametroJSON);
Get(url, fn_CargaFacturaCliente);
```

2.2 Llamadas con AJAX - Post

| | |
|--------------------|--------------------------------------------------------------------------------------------------------------------------------|
| Comentarios | Envío de información al servidor a través de un formulario con el método "Post" |
| Razón | <ul style="list-style-type: none"> Comunicación del Cliente con el servidor Envío de contenido seguro. |

```
// Ejemplo 1: Pasar parámetro de uno en uno.
const url='Cliente/Insert';
const form = new FormData();
form.append('cliente','Samuel');
form.append('distrito','SJL');

Post(url, form, fn_RegistrarCliente);

// Ejemplo 2: Pasar múltiples parámetros con un objeto
const url='Cliente/Insert';
const parametro= {cliente:'samu', distrito:'SJL'};
const form = new FormData();
form.append('par', JSON.stringify(parametro));

Post(url, form, fn_RegistrarCliente);
```

2.3 Llamadas con AJAX _Go_Url

| | |
|--------------------|-------------------------------------------------------------------------------------------------------------------------------------------------|
| Comentarios | Carga de una vista "página" del servidor con el método "Get" |
| Razón | <ul style="list-style-type: none"> Comunicación del Cliente con el servidor Cargar una página en el contenido principal |

```
const url = 'Administration/Ruta/EditRuta';
const parametro= 'idRuta:1';
_Go_Url(url, url, parametro);
```

3.0 HELPER – VALIDADORES

3.1 _rules

| | |
|-------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Comentarios | Asignación de reglas a los controles por eventos |
| Razón | <ul style="list-style-type: none"> Reducir líneas de código con reglas predefinidas. Uso básico e intermedio para formularios de mantenimiento. Asigna reglas para todos aquellos campos que tengan la clase "_enty" y que cumplan con los atributos de validación. |

```
// Uso del helper "_rules"
```

```
(function ini() {
  _rules({ id: 'panelOrdendeCompra', clase: '_enty' });
})();
```

```
// Plantilla de HTML con las reglas
```

```
<div class="col-sm-12" id="panelOrdenDeCompra">
  <div class="form-group">
    <label for="cboTipoPo" class="col-md-2 control-label">Type PO:</label>
    <div class="col-md-10">
      <select id="cboTipoPo" data-id="IdTipoPo" class="form-control _enty" data-min="1" data-max="20"
        data-required="true"></select>
    </div>
  </div>

  <div class="form-group">
    <label class="control-label col-sm-2">Client Price</label>
    <div class="col-sm-10">
      <input id="txtPrecioCliente" data-id="PrecioCliente" class="form-control _enty"
        type="text" data-min="1" data-max="12"
        data-required="true" data-type="dec" />
    </div>
  </div>

  <div class="form-group">
    <label class="control-label col-sm-2">Quantity</label>
    <div class="col-sm-10">
      <input id="txtCantidad" data-id="Cantidad" class="form-control _enty"
        type="text" data-min="1" data-max="12"
        data-required="true" data-type="int" />
    </div>
  </div>
</div>
```



```

<div class="form-group">
  <label class="control-label col-sm-2">Style</label>
  <div class="col-sm-10">
    <input id="txtStyle" data-id="StyleCode" class="form-control _enty"
      type="text" data-min="2" data-max="60"
      data-required="true" />
  </div>
</div>

<div class="form-group" id="grupoFechaArrivalPO">
  <label class="control-label col-md-2">Arrival PO:</label>
  <div class="col-md-10">
    <div class="input-group date">
      <input id="txtArrivalPO" data-id="ArrivalPo" class="form-control _enty"
        type="text" data-min="10" data-max="10"
        data-required="true" data-level="true" data-type="date">
      <div class="input-group-addon">
        <i class="fa fa-calendar"></i>
      </div>
    </div>
  </div>
</div>
</div>

```

3.2 _required

| | |
|-------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Comentarios | Asignación de campos requeridos a los controles. |
| Razón | <ul style="list-style-type: none"> Reducir líneas de código con reglas predefinidas. Uso básico e intermedio para formularios de mantenimiento. Asigna reglas de requeridos para todos aquellos campos que tengan la clase "_enty" y que cumplan con los atributos de validación. |

```

let reqOrdenDeCompra = _required({ id: 'panelOrdenDeCompra', clase: '_enty' });
if (!reqOrdenDeCompra) {
  swal({ title: "Mensaje", text: "Campos requeridos", type: "error" });
  return;
}

```

3.3 Empty

| | |
|-------------|---------------------------------------------------------------------------------------------------------------------------------------|
| Comentarios | Validar valores de vacíos, nulos o indefinidos. |
| Razón | <ul style="list-style-type: none"> Validar variables por valor y por tipo de dato, entre ellos cadena, objeto y array. |

```
// Validar Valores de tipo Cadena
let codigoEstilo = "";
_isEmpty(codigoEstilo); //respuesta => true

// Validar valores de tipo Array
let aEstilos = [];
_aIsEmpty(aEstilos); //respuesta => true


// Validar valores de tipo Objeto
let oEstilo = {};
_oIsEmpty(oEstilo); //respuesta => true
```

3.4 Parse

| | |
|-------------|---------------------------------------------------------------------------------------------------------------------------------|
| Comentarios | Validar y parsear los valores de tipo. |
| Razón | <ul style="list-style-type: none"> Validar valores por tipo de dato, entre ellos entero (int) y decimal (float). |

```
let edad= "20.15";
_parseInt(edad) // respuesta => 20

let monto= "20,500.25";
_parseFloat(monto); // respuesta => 20500.25
```

| | | |
|-----------------------------------------------------------------------------------|-------------------------------------------------|------------------|
|  | Estándares de Desarrollo para JavaScript | |
| | Dirección: Tecnología de la Información | |
| | Versión: 1.0 | Mayo 2019 |

3.5 CSVtoJSON

| | |
|--------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Comentarios | Es una función para convertir una variable de tipo cadena "CSV" a un array de objetos "JSON". |
| Razón | <ul style="list-style-type: none"> Reducir líneas de código y reutilizar el código predefinido. Reducir las tramas de respuesta con CSV. |

```
const aGrilla = CSVtoJSON(aPackingListCsv, '-', '^');

// CSVtoJSON(parametroCSV, parametroDelimitadorCampo, parametroDelimitadorFila)
```

3.6 _comboFromCSV

| | |
|--------------------|-----------------------------------------------------------------------------------------------------------------|
| Comentarios | Es una función para convertir una variable de tipo cadena "CSV" a un cadena de Options. "<option>....</option>" |
| Razón | <ul style="list-style-type: none"> Crear de forma rápida múltiples options desde un CSV. |

```
const optionsClientes = _comboFromCSV(clientesCSV);
```

3.7 _comboFromJSON


| | |
|--------------------|----------------------------------------------------------------------------------------------------------|
| Comentarios | Es una función para convertir una variable de tipo JSON a una cadena de Options. "<option>...</option>" |
| Razón | <ul style="list-style-type: none"> Crear de forma rápida múltiples options desde un JSON. |

```
const optionsClientes = _comboFromJSON(clientesJSON);
```

3.8 _comboItem

| | |
|--------------------|------------------------------------------------------------------------------------|
| Comentarios | Es una función para crear una cadena simple de option. "<option></option>" |
| Razón | <ul style="list-style-type: none"> Crear de forma rápida un option. |

```
const optionSeleccione = _comboItem({ value: '', text: '--Seleccione--' })
// Resultado => <option value="">--Seleccione--</option>
```

| | | |
|---------------------------------------------------------------------------------------------------------------------------|-------------------------------------------------|------------------|
|  <small>WORLD • TEXT • SOURCE</small> | Estándares de Desarrollo para JavaScript | |
| | Dirección: Tecnología de la Información | |
| | Versión: 1.0 | Mayo 2019 |

3.9 _convertDate_ANSI

| | |
|--------------------|-------------------------------------------------------------------------------------------------|
| Comentarios | Es una función para convertir una fecha del navegador al formato universal ANSI. |
| Razón | <ul style="list-style-type: none"> Utilizar la fecha ANSI para la base de datos. |

```
const fechaANSI = _convertDate_ANSI("04/30/2019")
// Resultado => 20190430
```

3.10 _getParameter

| | |
|--------------------|--------------------------------------------------------------------------------------------------------------------------------|
| Comentarios | Es una función que obtiene un objeto literal de todas las etiquetas que estén dentro de un contenedor con la clase "_enty". |
| Razón | <ul style="list-style-type: none"> Agrupar etiquetas y obtener valores de los campos requeridos del Formulario. |


```
const oParametros = _getParameter({ id: 'divEncabezado', clase: '_enty' });
/* Resultado =>
{
  fechaCliente : '20190430',
  codigoPO : 'PO-Fall-19',
  precio: 10.25
}
*/
```

3.11 _par

| | |
|--------------------|--------------------------------------------------------------------------------------|
| Comentarios | Es una función que busca y obtiene el valor de un parámetro de la página. |
| Razón | <ul style="list-style-type: none"> Obtener los valores de una página. |

```
// Div parametro
<div class="hide">
  <input type="text" id="txtParametroPO" value="idSolicitud:136,idEstado=0">
</div>
```

```
// Buscar y obtener el parametro
const txtParametro = document.getElementById('txtParametroPO').value;
const idSolicitud = _par(txtParametro,"idsolicitud");
// Resultado => 136
```

| | | |
|-----------------------------------------------------------------------------------|-------------------------------------------------|------------------|
|  | Estándares de Desarrollo para JavaScript | |
| | Dirección: Tecnología de la Información | |
| | Versión: 1.0 | Mayo 2019 |

3.12 _modalBody

| | |
|--------------------|---------------------------------------------------------------------------------------------------------|
| Comentarios | Es una función que muestra una ventana modal con una vista parcial. |
| Razón | <ul style="list-style-type: none"> Cargar una vista “pagina” dentro de una ventana modal |

```
_modalBody({
  url: 'Auditoria/Mensajeria/_EditarMensaje', //ruta de la vista
  ventana: 'editMensaje', //id ventana
  titulo: 'Editar Mensaje', //titulo ventana
  ancho: '1400', //ancho de la ventana en pixeles, opcional
  alto: '1000', //alto de la ventana en pixeles, opcional
  parametro: 'idMensaje:101'
})
```

3.13 swal

| | |
|--------------------|-----------------------------------------------------------------------------------|
| Comentarios | Es una función que muestra una ventana modal con mensajes personalizados. |
| Razón | <ul style="list-style-type: none"> Mostrar mensajes personalizados |

```
swal({
  title: 'Mensaje',
  text: 'Se actualizo correctamente',
  type: 'success'
})
```

4.0 ESTRUCTURA DEL ARCHIVO JAVASCRIPT

| | |
|--------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------|
| Comentarios | Todos los archivos JavaScript tendrán la estructura para mantener ordenado todas las funciones. |
| Razón | <ul style="list-style-type: none"> Encapsular y Ordenar todas las funciones de la pagina Acceder rápidamente a las funciones. |

```
// 1era parte : objeto Principal
var appEditarPO = (
  function (d, idPadre) {

    var oVariables =
    {
```

```

idgrupocomercial: '',
accion: '',
lstClienteTemporada: ''
}

//funcion de asignación de eventos para los controles y asignación de valores
function load(){
    // Asignación de eventos
    d.getElementById('btnActualizarPo').addEventListener('click', updatepo);

    // Asignación de valores
    const parametro = _('txtParametro').value;
    d.getElementById('txtCodigoPO').value = _par(parametro,'codigoPO');
}

//funcion de llamada al controlador por primera vez
function req_Inicio (){
    const parametro= d.getElementById('txtCodigoPO').value;
    const url='PO/POEstilo/getData_byPOProducto?par=';
    Get(url+parametro, res_Inicio);
}

//funcion de Respuesta
function res_Inicio(respuesta){
    let oRespuesta = !_isEmpty(respuesta) ? JSON.parse(respuesta) : null;
    ///...
}

//exponer funciones y variables
return
{
    load: load,
    req_ini: req_Inicio,
    ovariables: oVariables
}
}
)(document, 'panelEncabezadoPO');

// 2da parte : Funcion autoinvocable de inicio
(
    function ini() {
        appEditarPO.load();
        appEditarPO.req_Inicio();
    }
)();

```