Parte 1

Crud Javascript, css & sql.

Este es un tutorial de youtube el cual enseña cómo crear un crud mediante html css(boostrap) & Javascript(boostrap, sql, jquery)

Parte 1:

<https://www.youtube.com/watch?v=qVBAM-eBPh0&ab_channel=SergioGranizo>

Parte 2:

<https://www.youtube.com/watch?v=3Du8-C1mMGQ>

Parte 3:

<https://www.youtube.com/watch?v=3oVF0Jdq_S0&ab_channel=SergioGranizo>

Crearemos el documento html con sus respectivas dependencias:

<!DOCTYPE html>

<html lang="en">

  <head>

    <meta charset="UTF-8" />

    <meta http-equiv="X-UA-Compatible" content="IE=edge" />

    <meta name="viewport" content="width=device-width, initial-scale=1.0" />

    <title>Document</title>

    <link rel="stylesheet" href="style.css" />

    <link rel="stylesheet" href="boostrap/css/bootstrap.min.css" />

  </head>

  <body>

    <section class="seccion-dinamica-html">

      <h1 class="title-form">probando</h1>

      <div class="container">

        <div class="row">

          <form class="formulario-container">

            <fieldset>

              <legend>Descripcion</legend>

              <input type="text" class="form-control" id="item" />

              <br />

              <legend>Precio</legend>

              <input

                type="number"

                step="0.01"

                class="form-control"

                id="precio"

              />

*<!--––  <textarea name="comment" form="titleform">*

*Enter text here...</textarea*

*>––-->*

              <br />

              <button type="button" id="crear" class="btn btn-success">

                Crear Tabla

              </button>

              <button type="button" id="insertar" class="btn btn-success">

                Insertar

              </button>

              <button type="button" id="modificar" class="btn btn-danger">

                Modificar

              </button>

              <button type="button" id="listar" class="btn btn-success">

                Mostrar/Actualizar

              </button>

              <button type="button" id="borrarTodo" class="btn btn-danger">

                Eliminar

              </button>

            </fieldset>

          </form>

          <hr />

          <h3>Lista de productos</h3>

          <table

            class="table table-border table-hover"

            id="listaProductos"

          ></table>

        </div>

      </div>

    </section>

    <script src="jquery/jquery-3.6.0.min.js"></script>

    <script src="boostrap/js/bootstrap.min.js"></script>

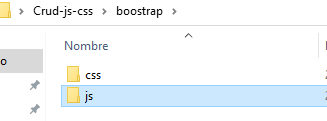
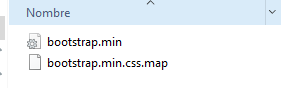
    <script src="crud.js"></script>

  </body>

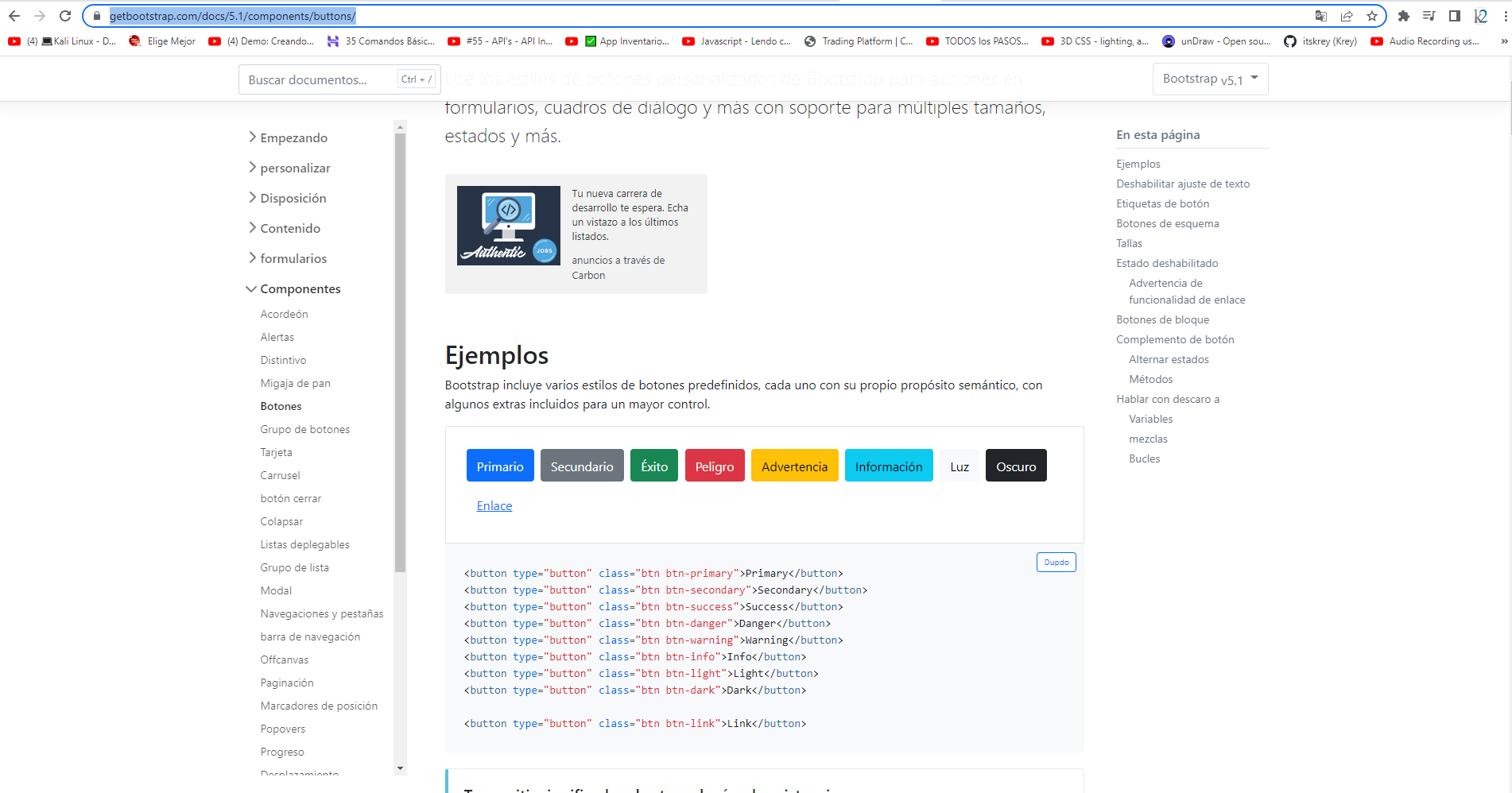
</html>

El min en el final de boostrap y jquery se refiere a que el código esta optimizado y utiliza solamente recursos necesarios es decir cargara mas rápido y es recomendable hacer esto.

<https://getbootstrap.com/docs/5.1/getting-started/download/>

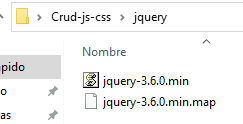
Esto es para descargar desde la página de boostrap los archivos locales e integrarlos con su mapa correspondiente.



Estos son los botones y aquí se llaman las clases para cambiarlos:

Todo está en la documentación de boostrap en su página principal.

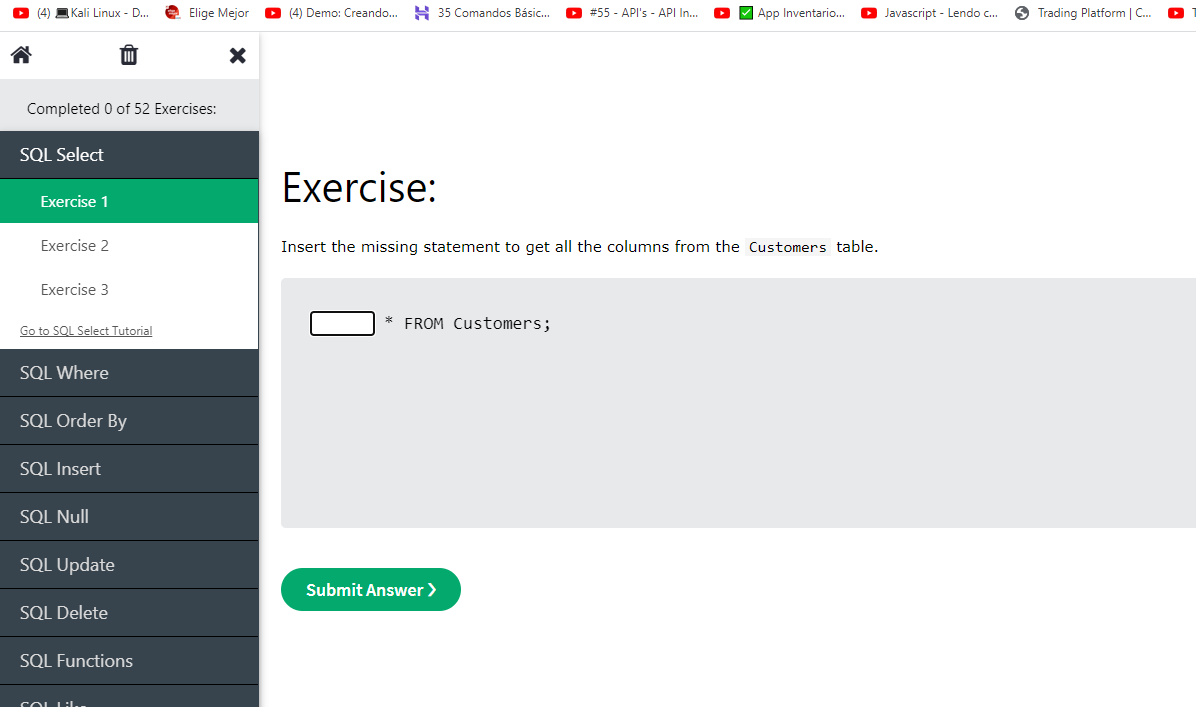
Muy bien ahora que tenemos la idea de cómo es boostrap utilizaremos jquery para no usar tanto código, al igual que boostrap jquery está en su respectiva documentación y la podremos descargar en su página principal

<https://jquery.com/download/>

Lo siguiente será aprender sql, el año pasado estudie mysql workbench y en realidad estudie lo básico lo cual me ayudo a entender esto de las bases de datos.

No utilizo muy seguido bases de dato y es por eso que preferí hacer esta documentación y para aprender a implementarla ya que esto si o si se tiene que utilizar en futuros proyectos.

<https://www.w3schools.com/sql/exercise.asp?filename=exercise_select1>



En estilos estableceremos los siguientes para que tenga un orden nuestra página:

\* {

  box-sizing: border-box;

  margin: 0;

  padding: 0;

}

.table {

  width: 715px;

}

.seccion-dinamica-html {

  width: 800px;

  margin: 0 auto;

}

.title-form {

  text-align: center;

}

.formulario-container {

  width: 750px;

}

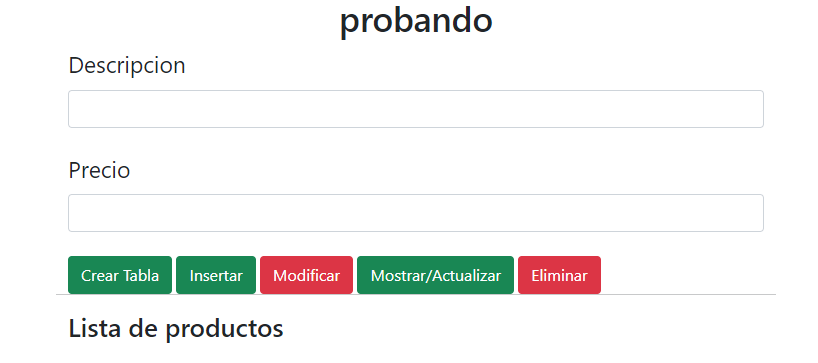
img {

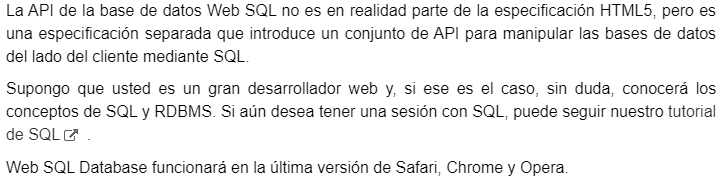
  width: 40px;

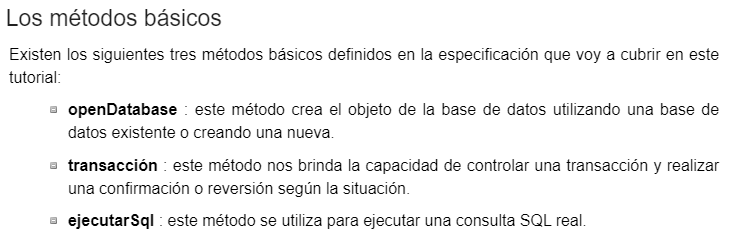
  height: 20px;

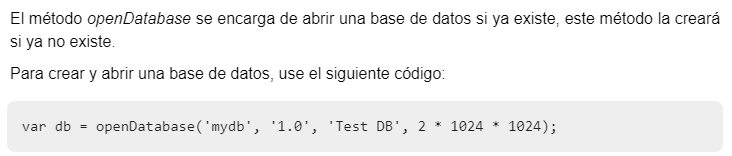
}

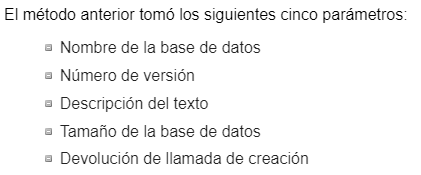
Resultado del código:

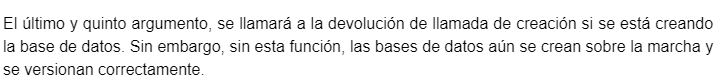


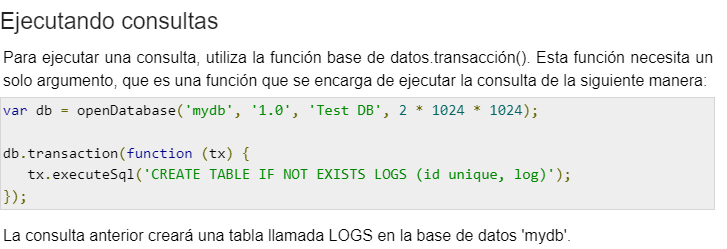
Conexión a la base de datos y conceptos clave:

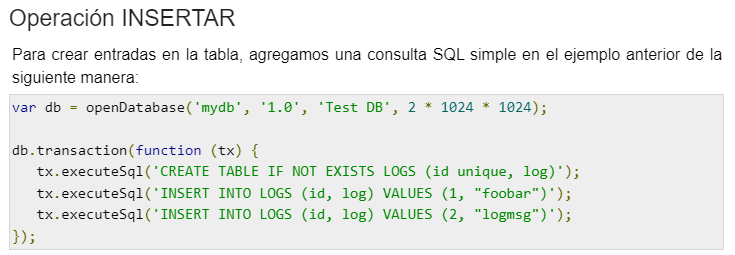


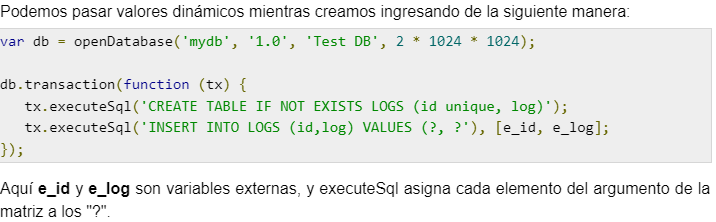








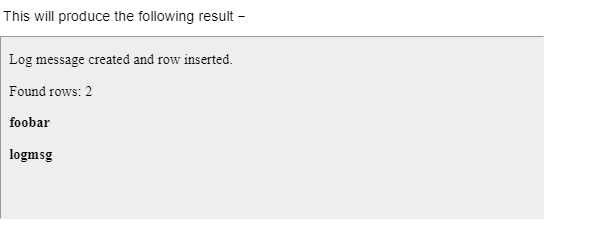












Para crear nuestra primera tabla utilizaremos el siguiente código

*/\**

*Nombre de la base de datos*

*Número de versión*

*Descripción del texto*

*Tamaño de la base de datos*

*Devolución de llamada de creación*

*\*/*

var nuevoID;

var db = openDatabase("itemDB", "1.0", "itemDB", 65535);

function limpiar() {

  document.getElementById("item").value = "";

  document.getElementById("precio").value = "";

}

*/\**

*openDatabase : este método crea el objeto de la base de datos utilizando*

*una base de datos existente o creando una nueva.*

*transaction : este método nos brinda la capacidad de controlar*

*una transacción y realizar una confirmación o reversión según la situación.*

*executeSql : este método se utiliza para ejecutar una consulta SQL real.*

*\*/*

$(function () {

  $("#crear").click(function () {

    db.transaction(function (transaction) {

      var sql =

        "CREATE TABLE products " +

        "(id INTEGER NOT NULL PRIMARY KEY AUTOINCREMENT, " +

        "item VARCHAR(100) NOT NULL, " +

        "precio DECIMAL(5,2) NOT NULL)";

      transaction.executeSql(

*//aqui hacemos la consulta de la transaccion sql*

        sql,

        undefined,

        function () {

          alert("Tabla creada satisfactoriamente");

        },

        function (transaction, err) {

          alert(err.message);

        }

      );

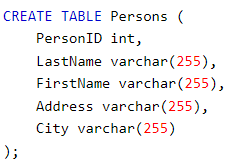
    });

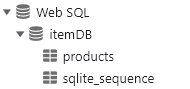
  });

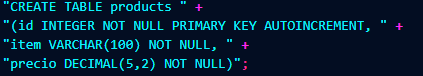
});

Este código es muy fácil de entender, lo único complicado es la tabla que a su ves podemos practicar:

<https://www.w3schools.com/sql/sql_create_table.asp>





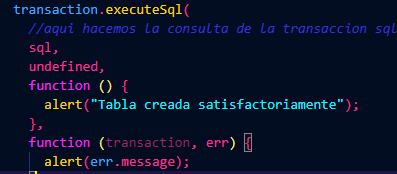


AQUÍ TENEMOS UNA DIFERENCIA DE **NOT NULL**: en SQL se usa para evitar insertar valores NULL en la columna especificada(row) considerándolo entonces como un valor no aceptado para esa columna.

**PRIMARY KEY**: en SQL son los campos que indican un identificado único del registro, para evitar que haya información duplicada en la tabla. Lo cual diferencia cada uno de los registros mediante la **primary key**.

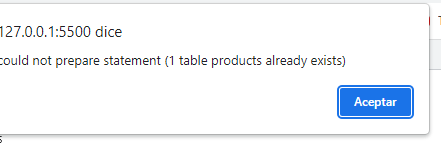
**AUTOINCREMENT**: es una función que opera en tipos de datos numéricos. Genera automáticamente valores numéricos secuenciales cada vez que se inserta un registro en una tabla para un campo definido como incremento automatico.

Entonces siempre es bueno poner un id en el cual se genere automáticamente e incremente su valor desde 1 a muchos.

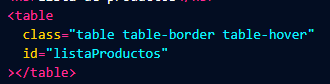


Este último código es muy esencial al crear la tabla ya que indicamos si fue creada satisfactoriamente o de lo contrario encontrara el error en sql.

 Ej: Si la tabla ya existe y le ponemos crear tabla entonces

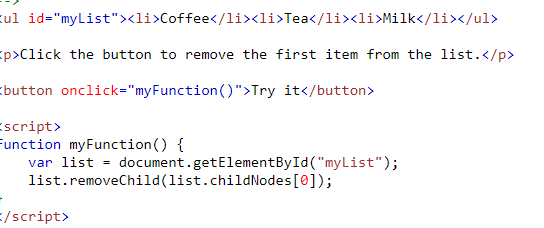
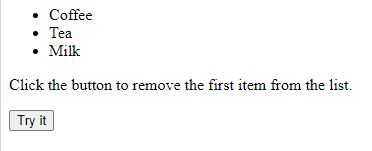


Este método limpia todo lo que tenga en lista de productos es decir

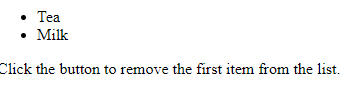


Todo lo que contenga dentro en caso de tener algún elemento entonces se eliminara completamente.

Aquí un ejemplo:

<http://www.w3bai.com/es/jsref/tryit.php?filename=tryjsref_node_removechild>

normal:

 Click en try it:

Eso es con js puro y elimina el index en la posición 0 o sea se elimina el primero luego se presiona otra vez pasa a ser el index que sigue que ahora es la posición del que se eliminó y así sucesivamente.

Este código está dentro de la primera función y lo siguiente será añadir una transacción llamando a la variable que contiene la base de datos.

Luego se creara una varibale que contendrá la consulta.

**SELECT**: selecciona

**\*** : hace referencia a todo

**FROM**: se quiere referir a una cosa o comúnmente (tabla)

**Productos**: es la tabla que creamos

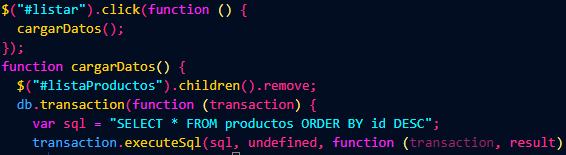
**ORDER**: La va a ordenar

**BY**: por

**Id:** el id

**DESC**: Descendentemente.





Este código de executeSql sirve para obtener un resultado desde la variable sql como parámetro que se creó afuera de la transacción que también es un parámetro.

Entonces aquí estamos comprobando si tenemos un numero de columnas con el resultado.

<https://www.w3schools.com/jsref/tryit.asp?filename=tryjsref_table_rows>

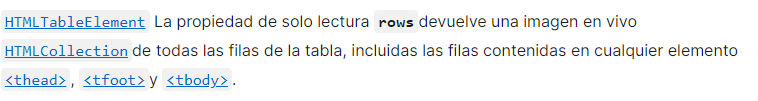
Comprueba cómo funciona rows.lenght

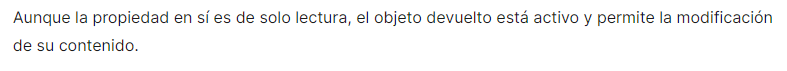
Entonces esto pregunta si tenemos datos dentro de la columna de rows

**Rows** = columnas

<https://developer.mozilla.org/en-US/docs/Web/API/HTMLTableElement/rows>

La propiedad de solo lecturas **row** devuelve una imagen en vivo de todas las filas de la tabla, incluidas las filas contenidas en cualquier elemento.



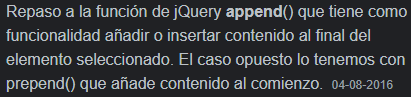


**Lenght** = es una propiedad de una función objeto, e indica el numero de argumentos que una función requiere, por ejemplo, el numero de paramtros formales

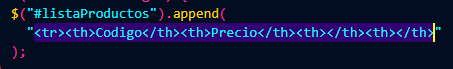
¿ Que es lenght ?

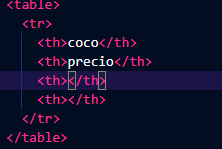
Se utiliza para manipular series de caracteres de todos los tipos de datos de serie de caracteres (BIT, BLOB y CHARACTER) y devuelve un valor de entero que proporciona el número de elemento de source\_string.





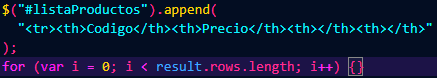
**Append**:

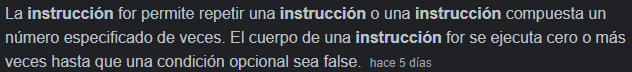


Esto es una estructura solamente de html y aquí realice la prueba:



Las 2 últimas etiquetas **th** son para darles espacios a los siguientes botones.

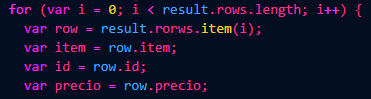
No recuerdo para que sirve el ciclo for:



Entonces tenenemos la variable **i** que es igual a 0;

Luego tenemos **i** si result es mayor;

Entonces entra **i++**

****

El ciclo for va a recorrer todo hasta que i sea mayor en este caso es para añadir un nuevo texto.