APIs

Plugins

{desafío} latam_

Crear plugins propios con jQuery para ampliar las funcionalidades de una aplicación web.

- Unidad 1:
  Introducción al lenguaje JavaScript
- Unidad 2:
  Funciones y Ciclos
- Unidad 3:
  Arrays y Objetos
- Unidad 4:
  APIs

Te encuentras aquí

{desafío}
latam_

¿Qué aprenderás en esta sesión?

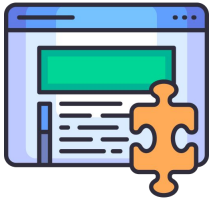
- *Crea un script que realice peticiones asíncronas utilizando la librería jQuery y AJAX para resolver un problema planteado.*

Entendiendo el Concepto de Plugin

- Los plugins son la utilidad que pone jQuery a disposición de los desarrolladores para ampliar las funcionalidades del framework. Por lo general, servirán para hacer cosas más complejas y necesarias para resolver necesidades específicas, pero las hacen de manera que puedan utilizarse en el futuro en cualquier parte y por cualquier web.

- En la práctica un plugin no es más que una función que se añade al objeto jQuery (objeto básico de este framework que devuelve la función jQuery para un selector dado), para que a partir de ese momento responda a nuevos métodos. Como ya sabemos, en este framework todo está basado en el objeto jQuery, así que con los plugins podemos añadirle nuevas utilidades.

- ¿Te imaginas creando tus propias capas de funcionalidades añadidas a jQuery?
- En este capítulo te explicaremos cómo diseñar tus propias soluciones y utilizar soluciones populares trabajadas por terceros.



Creación de un Plugin

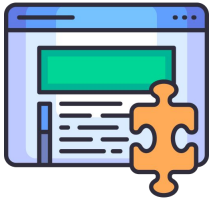
Reglas

Para poder crear un plugin es necesario aplicar y conocer ciertas reglas que son necesarias para respetar la estructura de un plugin como tal. Por ejemplo:

- El archivo que crees con el código de tu plugin lo debes nombrar como jQuery.[nombre de tu plugin].js. Por ejemplo jQuery.desaparece.js.

- Añade las funciones como nuevos métodos por medio de la propiedad fn del objeto jQuery, para que se conviertan en métodos del propio objeto jQuery.

- Dentro de los métodos que añades como plugins, la palabra "this" será una referencia al objeto jQuery que recibe el método. Por tanto, podemos utilizar "this" para acceder a cualquier propiedad del elemento de la página con el que estamos trabajando.



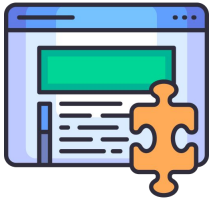
Creación de un Plugin

Reglas

- Debes colocar un punto y coma ";" al final de cada método que crees como plugin, para que el código fuente se pueda comprimir y siga funcionando correctamente. Ese punto y coma debes colocarlo después de cerrar la llave del código de la función.

- El método debe retornar el propio objeto jQuery sobre el que se solicitó la ejecución del plugin. Esto lo podemos conseguir con un return this; al final del código de la función.

- Se debe usar this.each para iterar sobre todo el conjunto de elementos que puede haber seleccionado. Recordemos que los plugins se invocan sobre objetos que se obtienen con selectores y la función jQuery, por lo que pueden haberse seleccionado varios elementos y no sólo uno. Así pues, con this.each podemos iterar sobre cada uno de esos elementos seleccionados. Esto es interesante para producir código limpio, que además será compatible con selectores que correspondan con varios elementos de la página.



Creación de un Plugin

Reglas

- Asigna el plugin siempre al objeto jQuery, en vez de hacerlo sobre el símbolo $, así los usuarios podrán usar alias personalizados para ese plugin a través del método noConfict(), descartando los problemas que puedan haber, si dos plugin tienen el mismo nombre.

Una vez conocidas estas reglas fundamentales, revisemos un ejemplo:

```
jQuery.fn.parpadea = function() {
   this.each(function(){
      elem = $(this);
      elem.fadeOut(250, function(){
         $(this).fadeIn(250);
      });
   });
   return this;
};
```

{desafío} latam_

Creación de un Plugin

Reglas

Ya tenemos nuestro primer plugin. Como puedes observar en el ejemplo, para generar una función del plugin debemos invocar la función jQuery.fn.[nombre_de_la_función] de esta forma jQuery entenderá que es una función diseñada por tí y propia. En particular esta función genera una especie de parpadeo suave en un elemento.

Bueno, ahora la queremos usar, entonces es sencillo:

```
    $(document).ready(function(){
       //parpadean los elementos de
class CSS "parpadear"
       $(".parpadear").parpadea();

       //añado evento click para un
botón. Al pulsar parpadearán los
elementos de clase parpadear

$("#botonparpadear").click(function()
{
          $(".parpadear").parpadea();
       })
    })
```

Ejercicio guiado - Creación de un plugin de datos financieros

{desafío}
latam_

Ejercicio guiado

Creación de un plugin de datos financieros

Desarrollar un plugin que permita consultar el valor de la UF, utilizando la API https://mindicador.cl/api de datos financieros. Utilizar el siguiente extracto del HTML:

```
<div id="miCaja">
   Valor de la UF
</div>
```

- **Paso 1**: Crear una carpeta en tu lugar de trabajo favorito y dentro de ella crea dos archivos, un index.html y un script.js.

Ejercicio guiado

Creación de un plugin de datos financieros

- **Paso 2:** En el index.html, escribir la estructura básica de un documento HTML, incorporar el enlace o CDN de jQuery, que nos permita incluir esta librería en nuestro ejemplo, agregar el extracto del código HTML entregado en el enunciado y enlaza el archivo externo denominado “script.js”, como se muestra a continuación:

```
<!DOCTYPE html>
<html lang="en">
<head>
   <meta charset="UTF-8">
   <meta name="viewport"
content="width=device-width,
initial-scale=1.0">
   <script
src="https://code.jquery.com/jquery-
3.7.1.min.js"></script>
   <title>Document</title>
</head>
<body>
   <div id="miCaja">
       Valor de la UF
   </div>
   <script src="script.js"></script>
</body>
</html>
```

Ejercicio guiado

Creación de un plugin de datos financieros

- **Paso 3:** Crear un archivo nuevo con el nombre de: jQuery.datosFinancieros.js dentro de la carpeta de trabajo, luego, crear el esquema básico y definir un nombre, por ejemplo datosFinancieros, que fue el nombre que se le dio al archivo en sí.

```
jQuery.fn.datosFinancieros = function() {
    return this;
};
```

Ejercicio guiado

Creación de un plugin de datos financieros

- **Paso 4**: Dentro de nuestro plugin llamar a la API como lo hicimos anteriormente (podemos copiar y pegar el código)

```
jQuery.fn.datosFinancieros = function() {
    $.ajax({
        type:"GET",
        url:"https://mindicador.cl/api",
        dataType:"json",
        success: function(data) {
      }
    });
   return this;
};
```

{desafío} latam_

Ejercicio guiado

Creación de un plugin de datos financieros

- **Paso 5:** Agregar el valor de la UF al elemento que use nuestro plugin y lo haremos a continuación:

```
jQuery.fn.datosFinancieros = function() {
  var element = this;
    $.ajax({
        type:"GET",
        url:"https://mindicador.cl/api",
        dataType:"json",
        success: function(data) {
              element.append(
                 `<span>${data.uf.valor}</span>`
              )
        }
    });
   return this;
};
```

{desafío} latam_

Ejercicio guiado

Creación de un plugin de datos financieros

- **Paso 6:** Creamos una variable element que le dimos el valor de this, esto debido a que si usamos this dentro de la función success, el this en ese contexto pertenece $.ajax y no a nuestro plugin, aclarado esto ahora podemos usar las bondades de jQuery para poder agregar un valor a nuestros elementos html. Ahora que tenemos nuestro plugin usémoslo llamándolo desde el archivo script.js. Mientras que en el archivo index.html, se debe agregar exactamente abajo de la inclusión del script.js, un nuevo enlace para el plugin mediante la instrucción: <script src="jQuery.datosFinancieros.js"></script>.

```
$(document).ready(function(){
   $('#miCaja').datosFinancieros();
});
```

Valor de la UF 28687.8

Integración de Plugins Externos basados en jQuery a un Proyecto

Muchos desarrolladores en el mundo, han aprovechado la potencia de jQuery para crear sus propios plugins basados en jQuery para generar proyectos más complejos y específicos. Estos proyectos complejos a su vez, los más populares siempre tienen soporte, son open source y abiertos para su uso. Permiten a equipos de desarrollo de pequeñas y medianas empresas, tener un punto de inicio para sus proyectos.

La integración de un plugin en un proyecto puede ser muy sencilla, la forma que recomendamos es que lo integres vía CDN (Content Delivery Network). En resumen, CDN es una url en donde el código de la librería es accesible a través de dicho link, de esa forma te aseguras que tu integración estará siempre en línea.

Es muy importante que antes de integrar el plugin tengas integrada la librería de jQuery, los script html los lee de forma secuencial, si no encuentra jQuery en tu proyecto antes que el plugin, tendrás errores al momento de ejecutar las funciones particulares que te ofrece el plugin.

Algunos Plugins comunes

Canvas JS

Este plugin es uno de los más populares porque te permite crear gráficas diversas, simplemente con añadir dicho plugin y la librería de jQuery. Por ende, se utilizó en el ejercicio anteriormente realizado. Por lo general, las librerías complejas que existen para generar gráficas son muy pesadas o requieren una instalación previa, tienden a sobrecargar el servidor en donde se encuentra alojada una aplicación web. En este sentido, CanvasJS es muy interesante puesto que es una librería ligera y sencilla de implementar.

Ejercicio guiado - CanvasJS

{desafío}
latam_

Ejercicio guiado

CanvasJS

Implementar la librería de CanvasJS con gráficos de columnas, con datos ya preestablecidos directamente en el código de JavaScript. Por lo tanto, en este caso el ejercicio solicita graficar el consumo de frutas tropicales en kilogramos por persona al año (datos ficticios), basados en la información suministrada por el ente encargado de llevar las estadísticas, siendo la información la siguiente: “Papaya: 23, Naranaja: 15, Platano: 25, Mango: 30 y Guayaba: 20”. A partir de la documentación facilitada por la página web CanvasJS, crear la estructura de la gráfica a utilizar usando el siguiente extracto de código HTML, extraído de la documentación de CanvasJS:

```
<div id="chartContainer" style="height: 300px; width: 100%;"></div>
```

Ejercicio guiado

CanvasJS

- **Paso 1:** Crear una carpeta en tu lugar de trabajo favorito y dentro de ella crea dos archivos, un index.html y un script.js.

- **Paso 2:** En el index.html, escribir la estructura básica de un documento HTML, incorporar el enlace o CDN de jQuery y CanvasJS, los cuales permitirán incluir estas librerías, luego, agregar el extracto del código HTML entregado en el enunciado y enlaza el archivo externo denominado “script.js”, como se muestra a continuación:

```
<!DOCTYPE html>
<html lang="en">
<head>
      <meta charset="UTF-8">
      <meta name="viewport"
content="width=device-width,
initial-scale=1.0">
      <title>CanvasJS</title>
</head>
<body>
      <div id="chartContainer"
style="height: 300px; width: 100%;"></div>
      <script
src="https://code.jquery.com/jquery-3.7.1.mi
n.js"></script>
      <script
src="https://canvasjs.com/assets/script/jque
ry.canvasjs.min.js"></script>
      <script src="script.js"></script>
</body>
</html>
```

Ejercicio guiado

CanvasJS

- **Paso 3:** Ahora en el archivo script.js, crear la estructura necesaria para poder graficar los datos indicados en el enunciado sobre el consumo de las frutas tropicales. En este caso, utilizamos como primera instrucción de JavaScript el evento load de windows, que permitirá ejecutar la función que le definamos cuando el documento quede cargado. Luego dentro de esta función, se captura el id del elemento HTML para poder activar el método correspondiente a CanvasJS, denominada CanvasJSChart, al cual le pasaremos toda la configuración necesaria para mostrar la gráfica.

```
window.onload = function () {
  $("#chartContainer").CanvasJSChart(options);
};
```

Ejercicio guiado

CanvasJS

- **Paso 4**: Crear las opciones que se le pasan al método, estas opciones serán un objeto con todos los datos, como el caso del título de la gráfica, el título y tamaño del título que tendrán los ejes de las abscisas y las ordenadas. Igualmente, el arreglo data que llevará el tipo de gráfico, en este caso column y los puntos a gráficas dataPoints, en donde el label es para las abscisas y las “y” para las ordenadas.

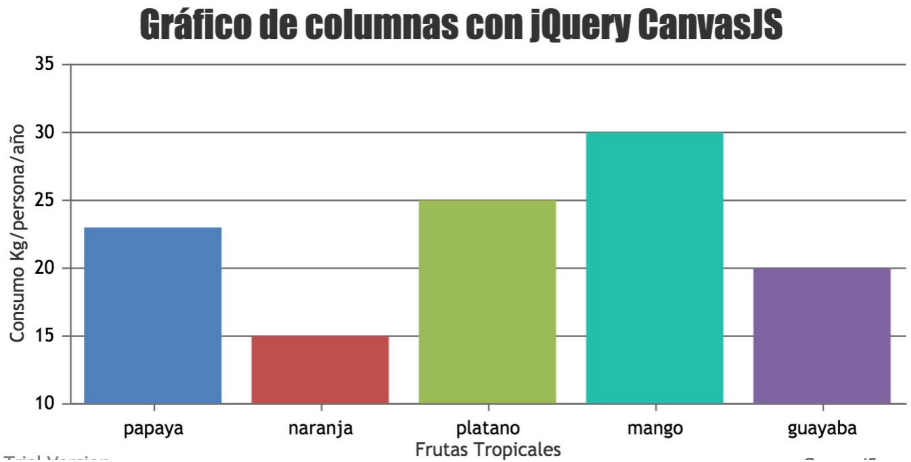
```
window.onload = function () {
  var options = {
       title: {
          text: "Gráfico de columnas con jQuery CanvasJS",
       },
       axisX:{
          title : "Frutas Tropicales",
          titleFontSize: 12
       },
       axisY:{
          title : "Consumo Kg/persona/año",
          titleFontSize: 12
       },
       data: [
          {
             type: "column",
             dataPoints: [
          { label: "papaya", y: 23 },
          { label: "naranja", y: 15 },
          { label: "platano", y: 25 },
          { label: "mango", y: 30 },
          { label: "guayaba", y: 20 },
             ],
          },
       ],
  };
  $("#chartContainer").CanvasJSChart(options);
};
```

{desafío} latam_

Ejercicio guiado

CanvasJS

- Paso 5: Al ejecutar el archivo index.html en el navegador web, encontraremos el siguiente resultado:



En el ejemplo anterior podemos observar una forma de aplicar la librería a un proyecto, si revisan la documentación oficial, notarán que sus gráficas se pueden customizar y añadir una enorme cantidad de parámetros en formato JSON.

Ejercicio guiado - jQuery, AJAX y CanvasJS

{desafío}
latam_

Ejercicio guiado

jQuery, AJAX y CanvasJS

Realizar un ejercicio implementado jQuery, AJAX y CanvasJS. El cual, consiste en mostrar la información del dolar según la API: https://mindicador.cl/api/dolar, en una gráfica dispuesta y configurada con CanvasJS, dicha configuración y documentación de la gráfica se puede encontrar en: jQuery Charts & Graphs from JSON Data Using AJAX. A partir de la documentación facilitada por la página web, se creará la estructura de la gráfica a utilizar. Para mostrar la gráfica, puedes usar el siguiente extracto de código HTML extraído de la documentación de CanvasJS:

```
<div id="chartContainer" style="height: 300px; width: 100%;"></div>
```

Ejercicio guiado

jQuery, AJAX y CanvasJS

- **Paso 1:** Crear una carpeta en tu lugar de trabajo favorito y dentro de ella crea dos archivos, un index.html y un script.js.

- **Paso 2:** En el index.html, escribir la estructura básica de un documento HTML, incorporar el enlace o CDN de jQuery y CanvasJS, los cuales, permitirán incluir estas librería en nuestro ejemplo, luego, agregar el extracto del código HTML entregado en el enunciado y enlaza el archivo externo denominado “script.js”, como se muestra a continuación:

```
<!DOCTYPE html>
<html lang="en">
<head>
   <meta charset="UTF-8">
   <meta name="viewport"
content="width=device-width,
initial-scale=1.0">
   <script
src="https://code.jquery.com/jquery-3.7.
1.min.js"></script>
   <script
src="https://canvasjs.com/assets/script/
jquery.canvasjs.min.js"></script>
   <title>Usando CanvasJS</title>
</head>
<body>
   <div id="chartContainer"
style="height: 300px; width:
100%;"></div>
   <script src="script.js"></script>
</body>
</html>
```

Ejercicio guiado

jQuery, AJAX y CanvasJS

- **Paso 3**: Ya ubicados en el archivo script.js, se debe incorporar la función de jQuery, luego dentro de ella, realizamos la conexión mediante AJAX a la API indicada y mostremos en consola los datos recibidos, en este caso se guardarán primero en una variable para futuras implementaciones.

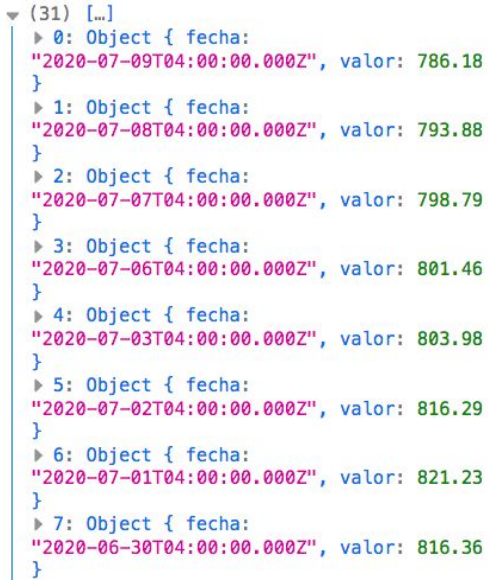
```
$(document).ready(function(){
     $.ajax({
        type:"GET",

url:"https://mindicador.cl/api/dolar",
        dataType:"json",
        success: function(datos) {
          let datosApi = datos.serie;
          console.log(datosApi);
        },
        error: function(error) {
          console.log(error)
        }
     });
});
```

Ejercicio guiado

jQuery, AJAX y CanvasJS

- **Paso 4:** Al ejecutar el código anterior y revisar la consola del navegador web, el resultado encontrado debería ser:



Ejercicio guiado

jQuery, AJAX y CanvasJS

- **Paso 5:** Ya comprobada la conexión con la API y que la información está llegando correctamente, pasamos a agregar la configuración para la gráfica, esta consiste en dos variables, una que será el arreglo que contenga la información extraída de la API mediante un ciclo repetitivo “for” y otra será un objeto con la configuración básica para la visualización de los datos en la gráfica, esta configuración es extraída de la documentación de CanvasJS :

```
var options =  {
        animationEnabled: true,
        theme: "light2",
        title: {
         text: "Daily Sales Data"
        },
        axisX: {
         valueFormatString: "DD MMM
YYYY",
        },
        axisY: {
         title: "USD",
         titleFontSize: 24,
        },
        data: [{
         type: "spline",
         dataPoints: dataPoints
        }]
    };
```

Ejercicio guiado

jQuery, AJAX y CanvasJS

Paso 5: Ahora solo queda cargar la información de la API dentro de la variable creada en el paso anterior del tipo arreglo y unir todo el código. Esto se puede lograr mediante una estructura repetitiva o ciclo for, lo cual, permitirá separar los datos exactos que necesita la gráfica, como son el valor del dólar y la fecha correspondiente a ese valor

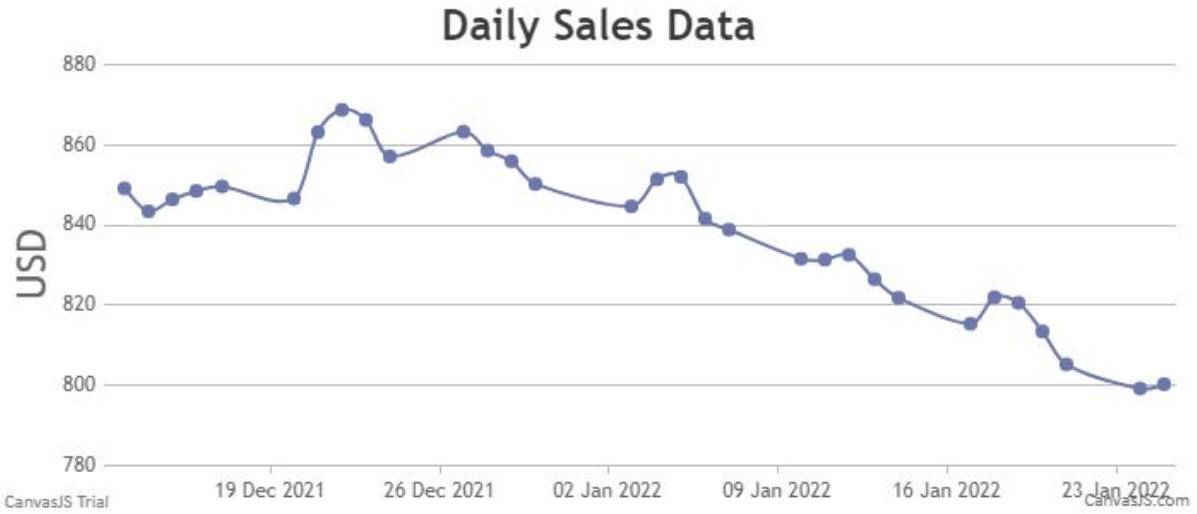
```
$(document).ready(function(){
    var dataPoints = []
    var options =  {
        animationEnabled: true,
        theme: "light2",
        title: {
         text: "Daily Sales Data"
        },
        axisX: {
         valueFormatString: "DD MMM YYYY",
        },
        axisY: {
         title: "USD",
         titleFontSize: 24,
        },
        data: [{
         type: "spline",
         dataPoints: dataPoints
        }]
    };
         $.ajax({
        type:"GET",
        url:"https://mindicador.cl/api/dolar",
        dataType:"json",
        success: function(datos) {
            let datosApi = datos.serie;
            for(var i = 0; i < datosApi.length; i++) {
                dataPoints.push({
                    x: new Date(datosApi[i].fecha),
                    y: datosApi[i].valor,
                })
            }
            $("#chartContainer").CanvasJSChart(options);
        },
        error: function(error) {
            console.log(error)
        }
         });
});
```

{desafío} latam_

Ejercicio guiado

jQuery, AJAX y CanvasJS

- **Paso 6:** Al ejecutar el código anterior, el resultado en nuestro navegador web será la gráfica con las tendencias del dólar a lo largo del tiempo:



Próxima sesión...

- *Desarrollaremos el material sincrónico que corresponde a una **Guía de ejercicios** con la cual practicarás todo lo aprendido en estas sesiones.*

{desafío}
latam_

Academia de
talentos digitales