This PDF contains Top 51 useful JavaScript Object Model (jsom) examples which can be used in SharePoint Online, SharePoint 2019/2016/2013 etc.

Top 51 SharePoint JavaScript Examples

Bhawana Rathore (Microsoft MVP)

Table of Contents

SharePointSky.com

SharePointSky.com

SharePointSky.com

About TSInfo Technologies:

TSInfo Technologies is one of the top SharePoint development & outsourcing company in India founded by two Microsoft MVPs. We have skilled certified professionals whose focus is to deliver high-quality work. We help organizations to build intranet to collaborate more effectively.

TSInfo Technologies works in technologies like SharePoint, Office 365, Azure, PowerBI, Nintex etc.

TSInfo Technologies services includes:

SharePoint development services	SharePoint consulting services
SharePoint migration services	SharePoint/Office 365 support service
Microsoft Azure consulting service	PowerBI consulting service
SharePoint Training Service	Nintex Training Service
SharePoint/Nintex corporate training service	SharePoint Apps or Add-in development
SharePoint customization/Branding service	SharePoint workflow, web part development

For any kind of SharePoint project work or corporate training contact us:

info@tsinfotechnologies.com || Mob: +91-9916854253 || https://www.TSInfoTechnologies.com

Sites we worked with:

https://www.EnjoySharePoint.com || https://www.SharePointSky.com

About Authors:

Bhawana Rathore:

Bhawana is a Microsoft MVP in Office Servers and Services category specialized in SharePoint. She has nearly 10 years of experience in Microsoft technologies like SharePoint, Office 365, SharePoint Online, Azure, PowerBI, Nintex etc. Bhawana is a SharePoint consultant having also migration experience in various tools like Sharegate, Metalogix, AvePoint etc. Bhawana also focuses on SharePoint technical blogging and blogs in EnjoySharePoint.com and SharePointSky.com.

SharePointSky.com

Introduction

In this tutorial, we are going to discuss various **SharePoint JavaScript examples**. The PDF contains the top 51 JavaScript SharePoint examples (**JSOM SharePoint examples**) which help you to learn the client object model.
The SharePoint jsom (JavaScript Object Model) examples will work in SharePoint Online and in SharePoint 2019/2016/2013.

JavaScript for SharePoint

In SharePoint Online, we use JavaScript, the popular client-side code to work with SharePoint objects.

JavaScript Object Model (JSOM) is very important in SharePoint nowadays because Microsoft is moving to the client-side and you should also use JSOM for new development.

To work with JSOM in SharePoint Online, it is better to have HTML, CSS, JavaScript, jQuery knowledge. Here I will give you a little introduction to jQuery so that you can at least use jQuery in SharePoint.

What is jQuery?

jQuery is a lightweight, “write less, do more”, JavaScript library.

The purpose of jQuery is to make it much easier to use JavaScript on your website.

jQuery takes a lot of common tasks that require many lines of JavaScript code to accomplish and wraps them into methods that you can call with a single line of code.

jQuery also simplifies a lot of the complicated things from JavaScript, like AJAX calls and DOM manipulation.

The jQuery library contains the following features:

- HTML/DOM manipulation
- CSS manipulation
- HTML event methods
- Effects and animations
- AJAX
- Utilities

When working with the SharePoint API via JavaScript it is essential to work with the JavaScript Object Model (JSOM).

JSOM is the use of JavaScript with the SharePoint client-side API which allows you to work with SharePoint without deploying code to the server itself. Due to SP Online relying on sand-boxed approaches, JSOM (or CSOM) is typically what would be used for code.

Will jQuery work in all browsers?

The jQuery team knows all about cross-browser issues, and they have written this knowledge into the jQuery library. jQuery will run exactly the same in all major browsers.

Editors For JavaScript/jQuery

Before knowing, how to start with JavaScript, we have to know what the available editors for JavaScript are. Some of the recommend editors for JavaScript are:

- Notepad
- Notepad++
- Sublime Text
- Visual Studio Code
- Atom
- BBEdit
- Komodo Edit
- Emacs etc.

Adding jQuery to Your Web Pages

There are several ways to start using jQuery on your web site. You can:

- Download the jQuery library from jQuery.com
- Include jQuery from a CDN, like Google, Microsoft or jQuery itself.

jQuery Google CDN

```
<script src="https://ajax.googleapis.com/ajax/libs/jquery/3.3.1/jquery.min.js"></script>
```

Refer jQuery Microsoft CDN

```
<script src="https://ajax.aspnetcdn.com/ajax/jQuery/jquery-3.3.1.min.js"></script>
```

jQuery CDN

```
<script src="https://code.jquery.com/jquery-3.4.0.min.js"></script>
```

Download jQuery

There are two versions of jQuery available for downloading:

- **Production version** – this is for your live/production website because it has been minified and compressed
- **Development version** – this is for testing and development (uncompressed and readable code)

Both versions can be downloaded from jQuery.com.

The jQuery library is a single JavaScript file, and you reference it with the HTML <script> tag.

```
<head>
<script src=”https://ajax.googleapis.com/ajax/libs/jquery/3.2.1/jquery.min.js”></script>
</head>
```

jQuery Syntax

The jQuery syntax is tailor-made for selecting HTML elements and performing some action on the element(s).

Basic syntax is:

- $(selector).action()
- A $ sign to define/access jQuery
- A (selector) to “query (or find)” HTML elements
- A jQuery action() to be performed on the element(s)

Examples:

- $(this).hide() – hides the current element.
- $(“p”).hide() – hides all <p> elements.
- $(“.test”).hide() – hides all elements with class=”test”.
- $(“#test”).hide() – hides the element with id=”test”.

SharePoint JavaScript Examples – PDF Download

You can download the **SharePoint JavaScript Examples** PDF at the end of the post.

What is JSOM in SharePoint Online?

JavaScript Object Model (JSOM) is a SharePoint Online client object model which is nothing but a set of classes and libraries. We can use those classes and objects to work with SharePoint data. To work with jsom, SP.js file should already be loaded in the page

Signup for an Office 365 SharePoint Business subscription

https://www.enjoysharepoint.com/sign-up-for-office-365-e5-free-trial/

How to use JSOM in SharePoint Online?

Jsom SharePoint code, we can use in:

- Content editor web part
- Script editor web part
- SharePoint hosted Add-in

JSOM has a few key elements to keep in mind

Work is done through a ClientContext object which acts as your window to operating with SharePoint. These are tied to a specific SP site and can be instantiated to the current site as well as a different site than the user is on.

When working with methods in the context to get such things as items or lists you will not be able to read the properties until the object is loaded. Likewise, operations are not executed until explicitly told to by calling **ClientContext.executeQueryAsync**.

The context works somewhat like a queue, operations are not completed until execute is called. Each executes represents a single call to the server. It is good practice to avoid calling execute until necessary.

To use jsom code inside a content editor web part and script editor web part in SharePoint, you can follow below steps:

How to use SharePoint JavaScript Code in Content Editor Web part

It is not recommended to write your HTML or css or jsom code directly inside the content editor web part. So first create an HTML File which will contain your HTML code and create .js file where we can write our JSOM code.

For creating these both Files, you can use Notepad++, Visual Studio or you can use visual studio code also.

Let's take an example, Create an HTML file, and add paragraph tag or content. Now, we can retrieve the data using JSOM and the JSOM code we can write inside the .JS file.

Below is the HTML File Code:

```
<!DOCTYPE html>
<html lang="en" xmlns="http://www.w3.org/1999/xhtml">
<script src="https://ajax.googleapis.com/ajax/libs/jquery/3.3.1/jquery.min.js"></script>
<script
src="https://onlysharepoint2013.sharepoint.com/sites/Raju/SiteAssets/Preetihtml/SpPro
perties.js"></script>
<head>
<meta charset="utf-8" />
<title></title>
</head>
<body>
<h2>Retrive Web Site Details</h2>
Site Title: <p id="pTitle"></p>
Site description: <p id="pdescript"></p>
Site template: <p id="ptemp"></p>
</body>
</html>
```

In this HTML Code, I added some paragraph tags for Site Title, Site description, and also Site template.

I have added different id for each paragraph like “pTitle”.

Below is the JavaScript object model (jsom) to get the SharePoint site details:

```
// JavaScript source code
ExecuteOrDelayUntilScriptLoaded(clickMethod, 'sp.js');
```

```
$(document).ready(function () {
});
var site;
function clickMethod() {
var clientContext = new SP.ClientContext.get_current();
site = clientContext.get_web();
clientContext.load(site);
clientContext.executeQueryAsync(success, failure);
}
function success() {
$("#pTitle").html(site.get_title());
$("#pdescript").html(site.get_description());
$("#ptemp").html(site.get_webTemplate());
}
function failure() {
alert("Failure!");
}
```

Developers write client-side code using the object model, but the operations are batched and sent as a single Extensible Markup Language (XML) request to the Client.svc service.

When the XML request is received, the Client.svc service makes calls to the server-side object model on behalf of the client. The results of the server-side calls are then sent back to the calling client in the form of a JavaScript Object Notation (JSON) object.

The SP.ClientContext class in the JavaScript client object model inherits from the SP.Client ContextRuntime class.

The ClientContextRuntime class provides two methods for loading objects: **Load** and **LoadQuery**.

The Load method specifies an object or collection to retrieve, while the LoadQuery method allows you to return collections of objects using a LINQ query.

Executing the Load or LoadQuery method does not cause the client to communicate with the server. Instead, it adds the load operation to a batch that will be executed on the server.

In fact, you may execute multiple load methods (as well as other operations) before calling the server. Each operation is batched, waiting for your code to initiate communication with the server.
To execute the batched operations, your code must call the ExecuteQuery or ExecuteQueryAsync method.

The **ExecuteQuery** method creates an XML request and passes it to the Client.svc service. The client then waits synchronously while the batch is executed and the JSON results are returned.

The **ExecuteQueryAsync** method, which is used in the Silverlight client object model, sends the XML request to the server, but it returns immediately. Designated success and failure callback methods receive a notification when the batch operation is complete.

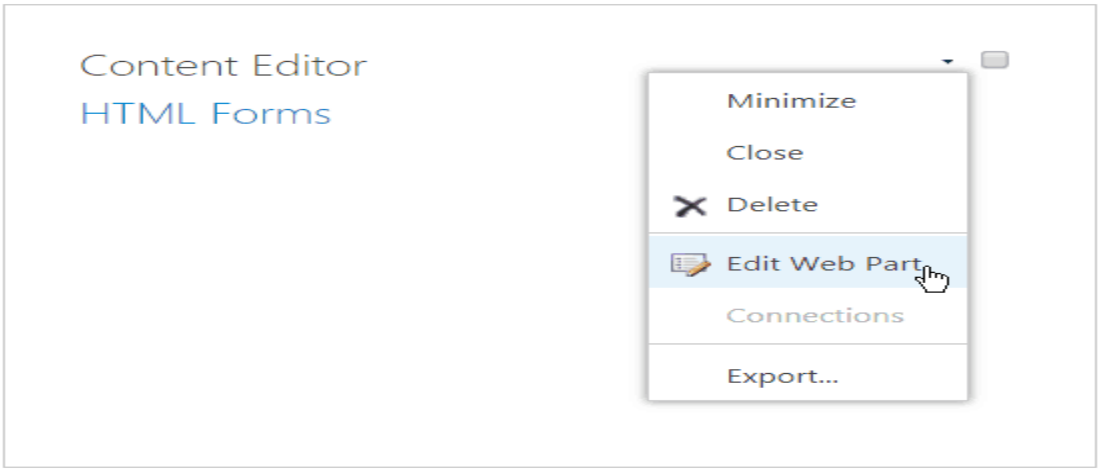
Here, by using this JSOM Code as **[“$(“#pTitle”).html(site.get_title());”]**, we can retrieve the Site Title data.

Similarly, we can retrieve the site description and site template.

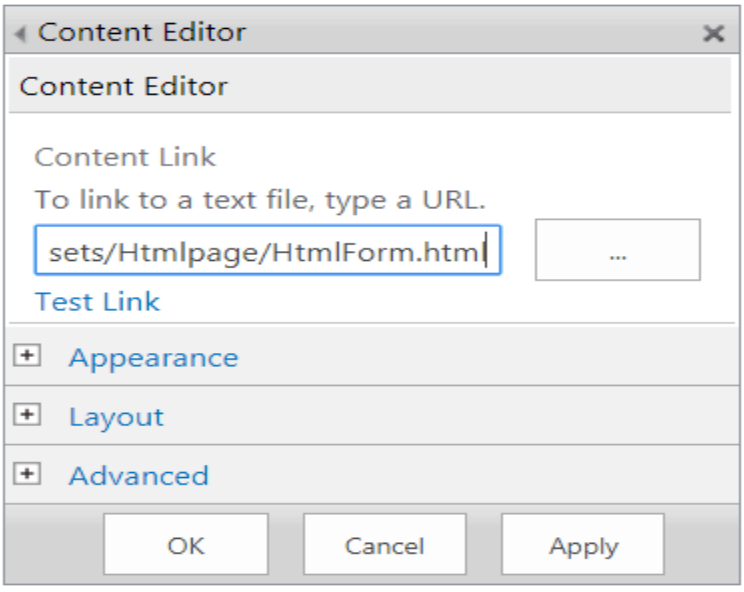
Add files into a content editor web part

Now, upload both the files (.HTML and .JS) to the SharePoint site assets library or any other library.

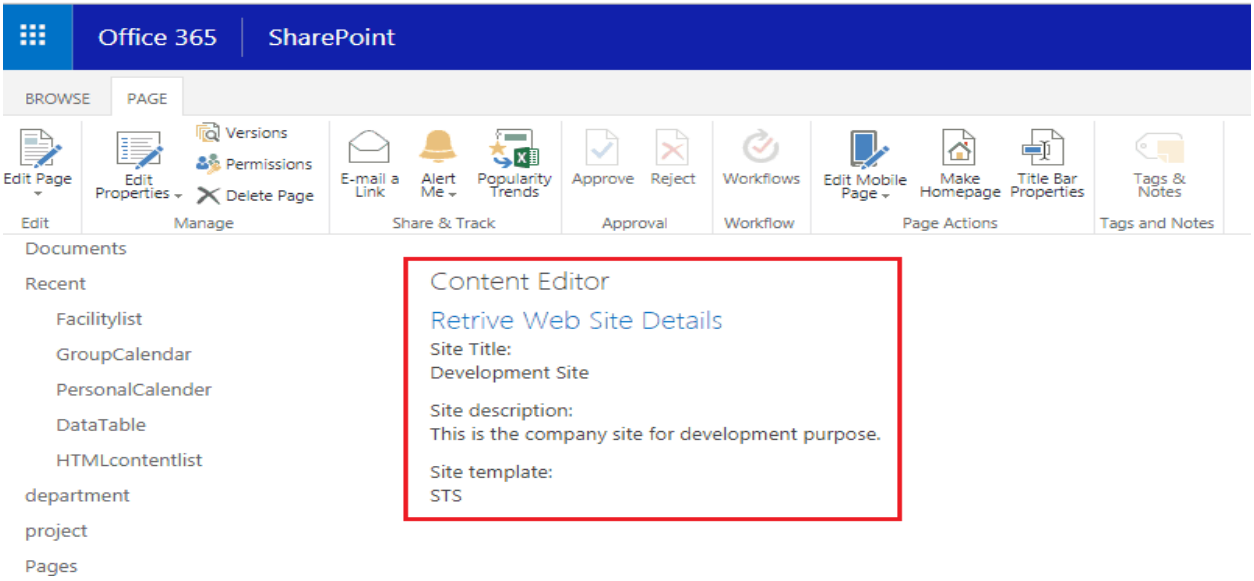
Then create a web part page, edit the web part page and then click on “Add a web part” to add a content editor web part to the page.
Once the content editor web part added, edit the web part like below:



Then in the web part properties dialog box, give your HTML File link path which you have uploaded inside the “Site Assets”. Then click on OK.



Now stop the editing, It will show you the final result as like given below screenshot.



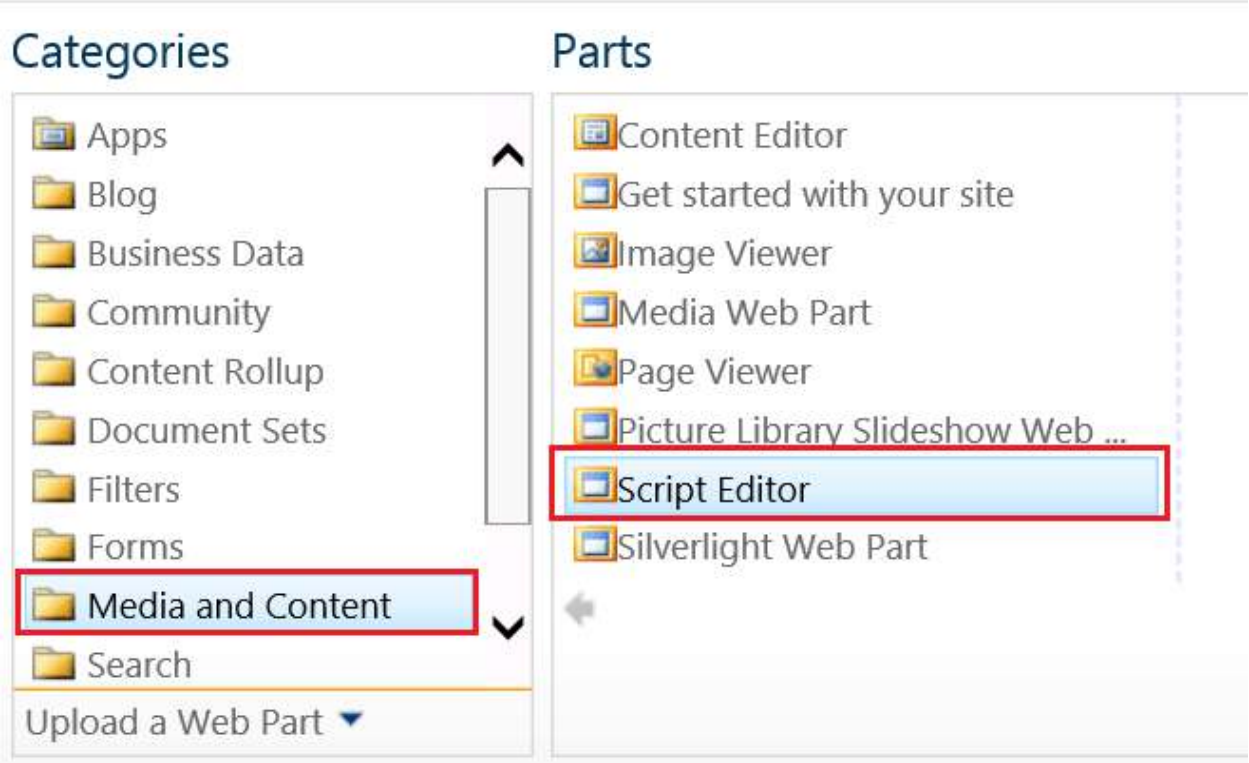
Here we can see the “Site Title”, “Site description” and “Site template”.

SharePoint JSOM Code in Script Editor Web part

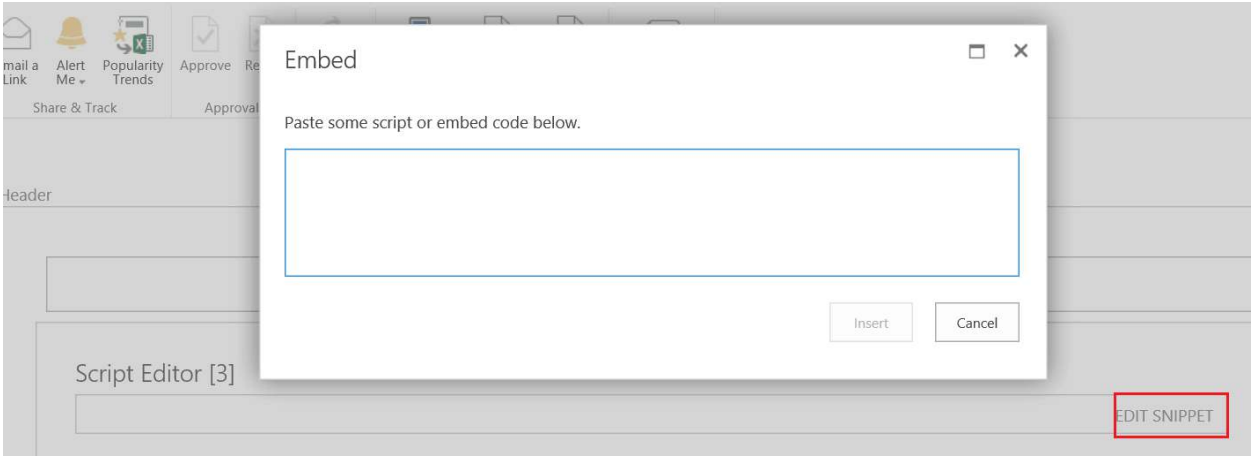
SharePoint 2013/2016 or SharePoint online provides script editor web part where we can directly write our js or HTML code.

To add a **script editor web part**, open the edit the page and then edit the web part page. Then click on **Add a web part** link. Then it will open the Web part categories in the ribbon.

From the web part categories, select **Media and Content** and then from the Parts select **Script Editor** and click on Add as shown in the fig below:



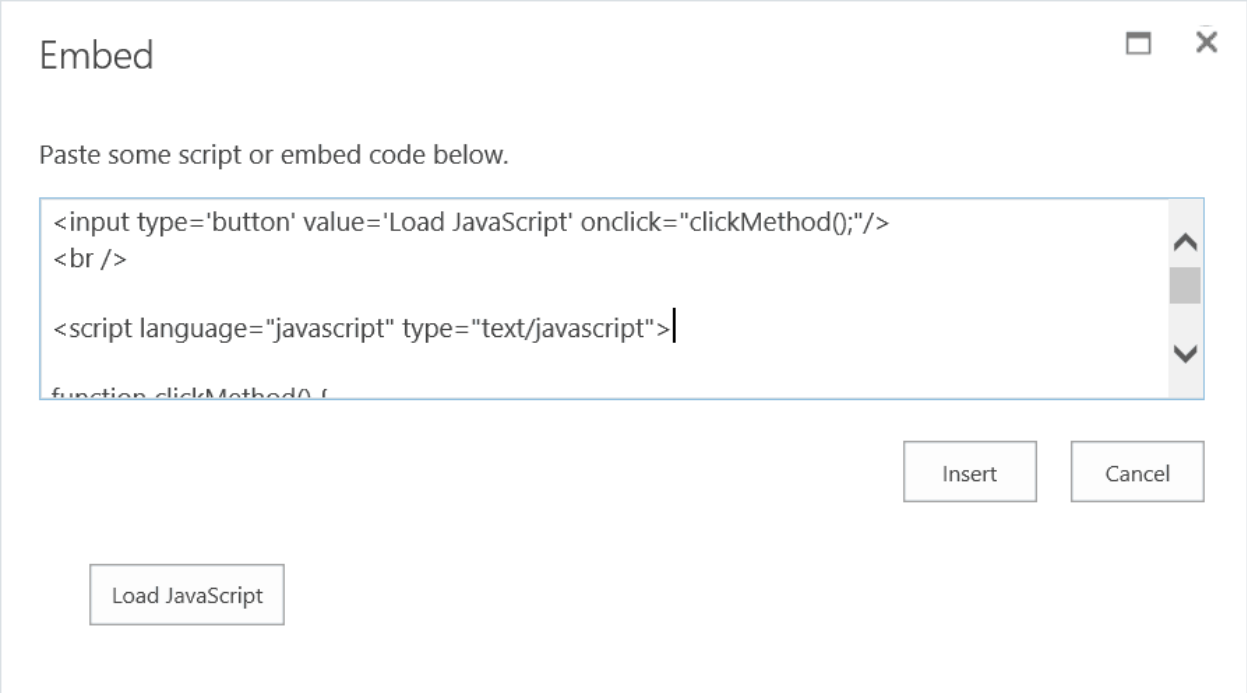
Once the web part added successfully click on Edit Snippet as shown in the fig below, then in the Embed section put your JavaScript code.



You can just add the below code to test if your javascript is working fine or not.

```
<input type='button' value='Load JavaScript' onclick="clickMethod();"/>
```

```
<br />
<script language="javascript" type="text/javascript">
function clickMethod() {
alert('hello');
}
</script>
```



Save the page and click on **Load JavaScript**, it should show you the alert which means it is working fine.

How to Call JavaScript Code in Button Click on SharePoint Online

We can use jQuery to call our jsom code in SharePoint Online on button click.

```
<script src="https://ajax.googleapis.com/ajax/libs/jquery/3.3.1/jquery.min.js"></script>
<script>
$(document).ready(function(){
$("#btnClick").click(function(){
```

```
yFunction();
});
});
function myFunction()
{
//Your jsom code will be here
var clientContext = new SP.ClientContext.get_current();
site=clientContext.get_web();
clientContext.load(site);
clientContext.executeQueryAsync(success, failure);
}
function success() {
alert(site.get_title());
}
function failure() {
alert("Failure!");
}
</script>
<input type='button' id='btnClick' value='Get Site Title'/>
```

Call jsom code in Page Load in SharePoint Online

SharePoint provides **ExecuteOrDelayUntilScriptLoaded()** method to call the jsom code after SP.js loaded successfully. Below is how we can a method on the page/form load after sp.js loaded successfully.

```
<script src="https://ajax.googleapis.com/ajax/libs/jquery/3.3.1/jquery.min.js"></script>
<script>
ExecuteOrDelayUntilScriptLoaded(myFunction,'sp.js');
function myFunction()
```

```
{
//Your jsom code will be here
var clientContext = new SP.ClientContext.get_current();
site=clientContext.get_web();
clientContext.load(site);
clientContext.executeQueryAsync(success, failure);
}
function success() {
alert(site.get_title());
}
function failure() {
alert("Failure!");
}
</script>
```

This way, we will be able to run the JSOM code on page load in SharePoint Online or SharePoint 2013/2016/2019.

General JSOM

These are general examples on some common snippets or patterns when working with JSOM objects

Object must support getEnumerator function

Enumerating Through Collections

```
var enumerator = yourObjCollection.getEnumerator();
while (enumerator.moveNext()) {
var curObj = enumerator.get_current();
```

```
//i.e. alert(curItem.get_title());
}
```

Targeting Current Web

```
//Targets the SPWeb the user is currently browsing
var ctx = new SP.ClientContext.get_current();
```

Targeting Specific Web

```
//Targets the SPWeb via URL (Server Relative or Absolute)
var ctx = new SP.ClientContext("/sites/yoursite");
//Use '/' for top level SPWeb at root of domain
var rootDomainWeb = new SP.ClientContext("/");
//For the root web of the current site collection leverage _spPageContextInfo
var curSiteColRootCtx = new
SP.ClientContext(_spPageContextInfo.siteServerRelativeUrl);
```

Top 51 SharePoint JavaScript Examples

Below are the top 51 SharePoint JavaScript Examples or JSOM SharePoint Online examples.

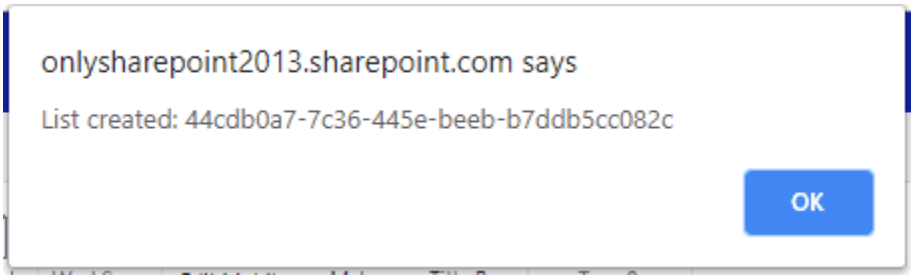
Example-1: Create List using JavaScript Object Model (JSOM) in SharePoint Online

This JSOM SharePoint examples explain, how to create a list using jsom (JavaScript Object Model) in SharePoint Online Office 365. The same jsom SharePoint code you can also create a list in SharePoint 2019/2016/2013.

```
<script src="https://ajax.googleapis.com/ajax/libs/jquery/3.2.1/jquery.min.js"></script>
<script>
```

```
$(document).ready(function() {
SP.SOD.executeFunc('sp.js', 'SP.ClientContext', listCreation);
});
function listCreation ()
{
var ctx = new SP.ClientContext.get_current();
var curWeb = ctx.get_web();
var listCreationInfo = new SP.ListCreationInformation();
listCreationInfo.set_title('My Custom List');
//To view all list types, view the SP.ListTemplateType enumeration on a SharePoint
Website
listCreationInfo.set_templateType(SP.ListTemplateType.genericList);
var myList = curWeb.get_lists().add(listCreationInfo);
ctx.load(myList);
ctx.executeQueryAsync(
function(){ alert('List created: ' + myList.get_id().toString()); },
function(sender, args){ alert('Error: ' + args.get_message()); }
);
}
</script>
```

Once you run the above code, it will show a message like below:



Now go to the site contents page of your SharePoint online and you will be able to see list created.

This is the SharePoint Online modern experience site contents page.
https://www.sharepointsky.com/sharepoint-modern-experience/
https://www.sharepointsky.com/sharepoint-online-switch-to-modern-experience/

Example-2: Update SharePoint Online List Title using JavaScript

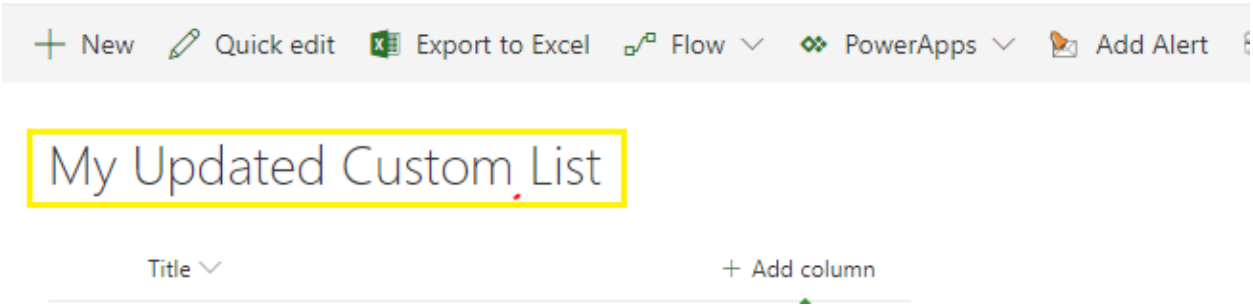
This jsom SharePoint example, we will see how to update list title using jsom in SharePoint Online.

Here we are updating the list name from '**My Custom List**' to '**My Updated Custom List**' using JSOM in SharePoint.

```
<script src="https://ajax.googleapis.com/ajax/libs/jquery/3.2.1/jquery.min.js"></script>
<script>
$(document).ready(function() {
SP.SOD.executeFunc('sp.js', 'SP.ClientContext', listUpdation);
});
function listUpdation ()
{
var ctx = new SP.ClientContext.get_current();
var curWeb = ctx.get_web();
var myList = curWeb.get_lists().getByTitle('My Custom List');
myList.set_title("My Updated Custom List");
myList.update();
```

```
ctx.load(myList);
ctx.executeQueryAsync(
function(){ alert('List name updated!'); },
function(sender, args){ alert('Error: ' + args.get_message()); }
);
}
</script>
```

Once you run the above code, you can see a confirmation message that list title updated successfully. Open the list to view the updated list title.



Note: The above code will only change the SharePoint list title, not the list URL.

Example-3: Delete List using JSOM (JavaScript Object Model) in SharePoint Online

These SharePoint examples explain we will see how to delete a SharePoint list using jsom (JavaScript Object Model). Here we are deleting the list name 'My Updated Custom List 'using JSOM.

```
<script src="https://ajax.googleapis.com/ajax/libs/jquery/3.2.1/jquery.min.js"></script>
<script>
$(document).ready(function() {
SP.SOD.executeFunc('sp.js', 'SP.ClientContext', listDeletion);
```

```
});
function listDeletion()
{
var ctx = new SP.ClientContext.get_current();
var curWeb = ctx.get_web();
var myList = curWeb.get_lists().getByTitle('My Updated Custom List');
myList.deleteObject();
ctx.executeQueryAsync(
function(){ alert('List deleted!'); },
function(sender, args){ alert('Error: ' + args.get_message()); }
);
}
</script>
```

Once you run the code, it will display an alert that the List Deleted.

Example-4: Create Field or Column in SharePoint List using JavaScript

These SharePoint examples explain, how to add one multiline column to the SharePoint list "**MyList**" using jsom which will be available with the default content type.

```
<script src="https://ajax.googleapis.com/ajax/libs/jquery/3.2.1/jquery.min.js"></script>
<script>
$(document).ready(function() {
SP.SOD.executeFunc('sp.js', 'SP.ClientContext', createField);
});
function createField ()
{
var ctx = new SP.ClientContext.get_current();
```

```
var curWeb = ctx.get_web();
var myList = curWeb.get_lists().getByTitle('MyList');
ctx.load(myList);
ctx.executeQueryAsync(
function(){
//Create the field
var fields = myList.get_fields()
fields.addFieldAsXml('<Field ID="{c10f042e-ab13-435c-ab93-9885f0d48d02}"
Name="myInternalName" DisplayName="A Multi User Column" Group="GroupName"
Type="UserMulti" Mult="TRUE"></Field>', true,
SP.AddFieldOptions.addToDefaultContentType)
ctx.executeQueryAsync(
function (){ alert('Created field!'); },
function (){ alert('Error: ' + args.get_message()); }
);
},
function(sender, args){ alert('Error: ' + args.get_message()); }
);
}
</script>
```

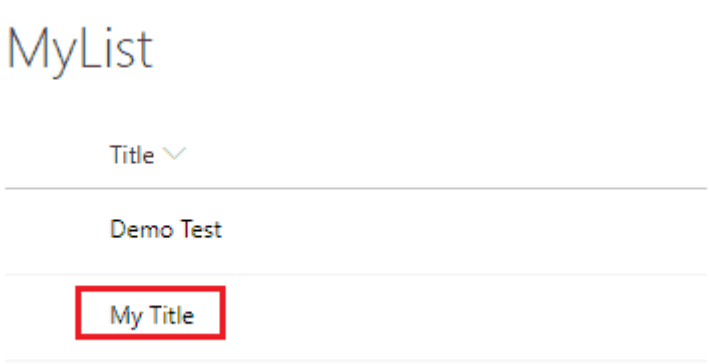
Below is the out where we have created one field named “**A Multi User Column**” in a MyList using JSOM in SharePoint Online.

Example-5: Add item to SharePoint list using JavaScript Object Model (jsom)

These SharePoint examples explain, how to add an item to the SharePoint Online list “MyList” using JSOM (JavaScript Object Model).

```
<script src="https://ajax.googleapis.com/ajax/libs/jquery/3.2.1/jquery.min.js"></script>
<script>
$(document).ready(function() {
SP.SOD.executeFunc('sp.js', 'SP.ClientContext', addListItem);
});
function addListItem ()
{
var ctx = new SP.ClientContext.get_current();
var customList = ctx.get_web().get_lists().getByTitle('MyList');
var itemCreateInfo = new SP.ListItemCreationInformation();
var listItem = customList.addItem(itemCreateInfo);
listItem.set_item('Title', 'My Title');
listItem.update();
ctx.load(listItem);
ctx.executeQueryAsync(
function(){ alert('Item created: ' + listItem.get_id()); },
function(sender, args){ alert('Error: ' + args.get_message()); }
);
}
</script>
```

Here we have added an item into the SharePoint list using jsom, you can see the output like below:



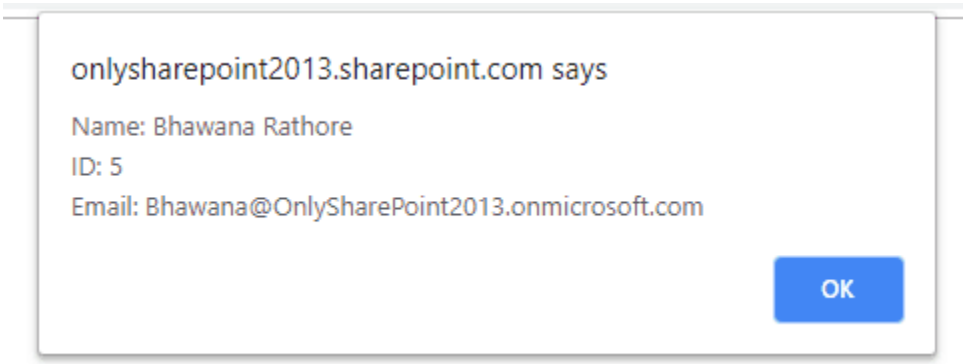
Example-6: Get user information from Person or Group column in SharePoint List using JSOM

Here we have a list called MyList and there we are fetching user information from the column Person Or Group by JSOM (JavaScript Object Model), which is a people picker type column.

```
<script src="https://ajax.googleapis.com/ajax/libs/jquery/3.2.1/jquery.min.js"></script>
<script>
$(document).ready(function() {
SP.SOD.executeFunc('sp.js', 'SP.ClientContext', getUser);
});
function getUser ()
{
var itemId = 1;
var ctx = new SP.ClientContext.get_current();
var customList = ctx.get_web().get_lists().getByTitle('MyList');
var listItem = customList.getItemById(itemId);
ctx.load(listItem);
ctx.executeQueryAsync(
function(){
```

```
//People Picker Multi OR Single value
var usrOutput = [];
var usrGrps = listItem.get_item("PersonOrGroup");
//Multi fields return arrays, singles do not. Unify to produce array in either case.
var userItems = [].concat( usrGrps );
userItems.forEach(function(userOrGroup){
usrOutput.push("Name: " + userOrGroup.get_lookupValue() + "\nID: " +
userOrGroup.get_lookupId() + "\nEmail: " + userOrGroup.get_email());
});
alert(usrOutput.join("\n\n"));
},
function(sender, args){ alert('Error: ' + args.get_message()); }
);
}
</script>
```

Here, you can see it displays the user information like user name, id, and email et, from the PersonOrGroup column.

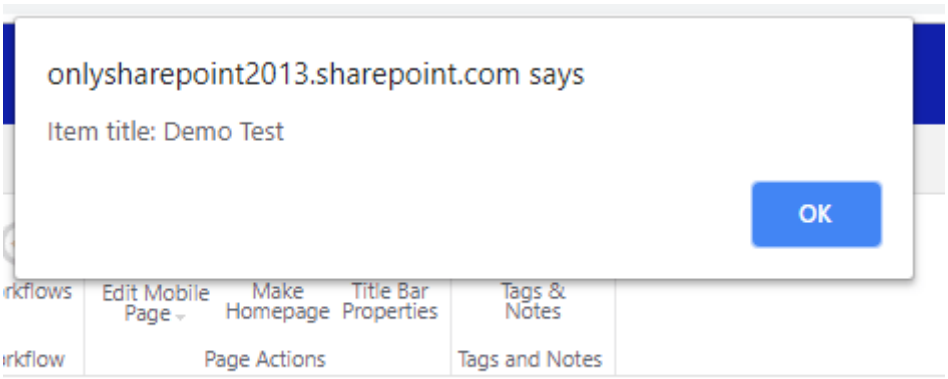


Example-7: Retrieve List Item by ID using JSOM (JavaScript Object Model) SharePoint Online

Now, we will see how to retrieve list item by list item id from the SharePoint list name MyList based on an item id using JSOM.

```
<script src="https://ajax.googleapis.com/ajax/libs/jquery/3.2.1/jquery.min.js"></script>
<script>
$(document).ready(function() {
SP.SOD.executeFunc('sp.js', 'SP.ClientContext', geyByID);
});
function geyByID ()
{
var itemId = 1;
var ctx = new SP.ClientContext.get_current();
var customList = ctx.get_web().get_lists().getByTitle('MyList');
var listItem = customList.getItemById(itemId);
ctx.load(listItem);
ctx.executeQueryAsync(
function(){
alert('Item title: ' + listItem.get_item("Title"));
},
function(sender, args){ alert('Error: ' + args.get_message()); }
);
}
</script>
```

Once you run the above code, you can see it will display Title column value of Item whose ID=1.



Here I have hard-coded the item id as 1, you can take a textbox and make this thing dynamic.

Example-8: Get List Item by Item ID by using CAML

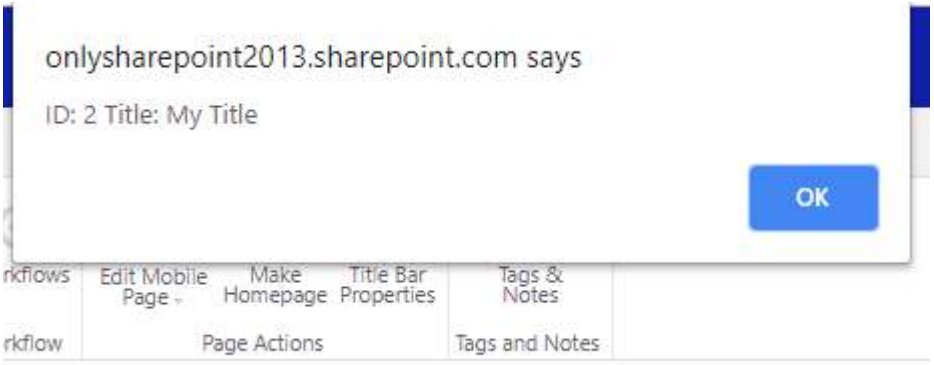
In this jsom SharePoint example I will show you how to retrieve list item by id using CAML in JavaScript Object Model in SharePoint Online.

Here we are fetching an item from the list name MyList by using the SharePoint CAML query.

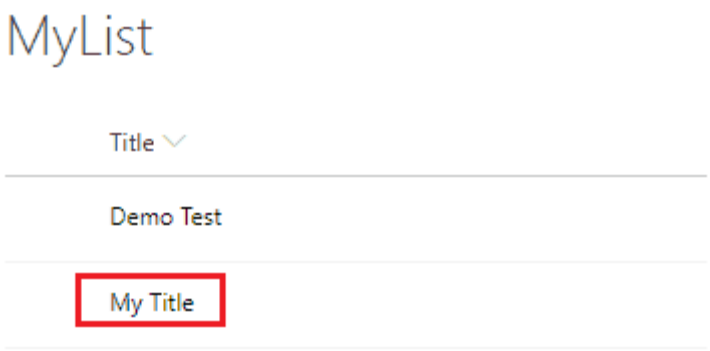
```
<script src="https://ajax.googleapis.com/ajax/libs/jquery/3.2.1/jquery.min.js"></script>
<script>
$(document).ready(function() {
SP.SOD.executeFunc('sp.js', 'SP.ClientContext', geyByCAML);
});
function geyByCAML ()
{
var ctx = new SP.ClientContext.get_current();
var customList = ctx.get_web().get_lists().getByTitle('MyList');
var camlQuery = new SP.CamlQuery();
```

```
camlQuery.set_viewXml('<View><Query><Where><Geq><FieldRef
Name=\'ID\'/><Value
Type=\'Number\'>2</Value></Geq></Where></Query><RowLimit>10</RowLimit></Vie
w>');
var listItemCol = customList.getItems(camlQuery);
ctx.load(listItemCol);
ctx.executeQueryAsync(
function(){
var listItems = [];
var listEnum = listItemCol.getEnumerator();
while (listEnum.moveNext()) {
var item = listEnum.get_current();
listItems.push('ID: ' + item.get_id() + ' Title: ' + item.get_item('Title'));
}
alert(listItems.join("\n"));
},
function(sender, args){ alert('Error: ' + args.get_message()); }
);
}
</script>
```

Once you run the above jsom SharePoint code, you can see the output like below:



You can see the list which has two items like below:



Example-9: Update SharePoint List Item using JSOM (JavaScript Object Model) in SharePoint Online

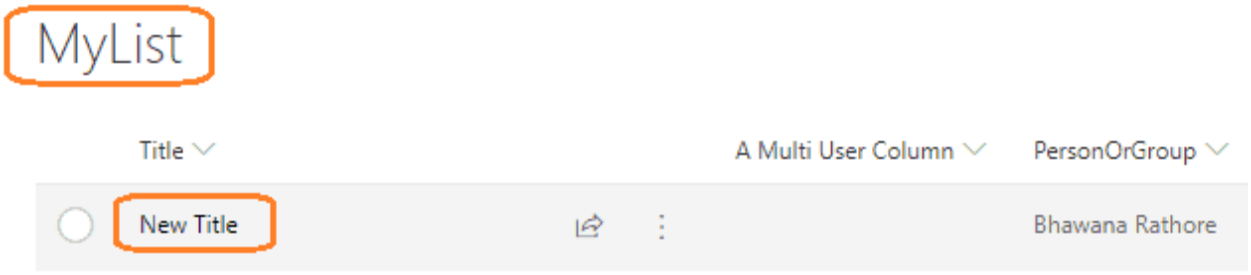
Here in this SharePoint javascript example, I will show, how to update the list item using the JavaScript object model (jsom) in SharePoint Online.

We will update the SharePoint list item by item id using JavaScript.

```
<script src="https://ajax.googleapis.com/ajax/libs/jquery/3.2.1/jquery.min.js"></script>
<script>
$(document).ready(function() {
SP.SOD.executeFunc('sp.js', 'SP.ClientContext', listItemUpdation);
```

```
});
function listItemUpdation ()
{
var itemId = 1;
var ctx = new SP.ClientContext.get_current();
var customList = ctx.get_web().get_lists().getByTitle('MyList');
var listItem = customList.getItemById(itemId);
listItem.set_item('Title', 'New Title');
listItem.update();
ctx.executeQueryAsync(
function(){ alert('Item updated');
})
}
</script>
```

Once you run the code, you can see the title will be changed to “New Title”.



Example-10: Delete List Item using JavaScript Object Model (jsom) SharePoint

Now we can see how to delete a list item using jsom in SharePoint Online.

Here we will delete an item from the list name MyList based on item id using JSOM (JavaScript Object Model).

```
<script src="https://ajax.googleapis.com/ajax/libs/jquery/3.2.1/jquery.min.js"></script>
<script>
$(document).ready(function() {
SP.SOD.executeFunc('sp.js', 'SP.ClientContext', listItemDeletion);
});
function listItemDeletion ()
{
var itemId = 1;
var ctx = new SP.ClientContext.get_current();
var customList = ctx.get_web().get_lists().getByTitle('CustomList');
var listItem = customList.getItemById(itemId);
listItem.deleteObject();
ctx.executeQueryAsync(
function(){ alert('Item deleted: ' + itemId); },
function(sender, args){ alert('Error: ' + args.get_message()); }
);
}
</script>
```

Once you execute the above code, the list item with ID=1 will be get deleted from the SharePoint Online list.

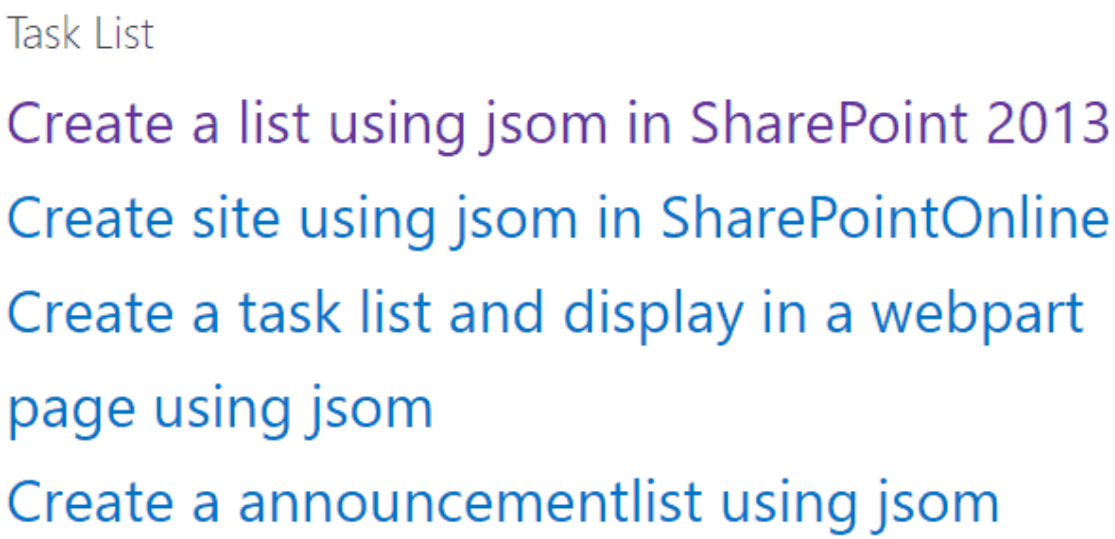
Example-11: Retrieve All List Items from SharePoint List using JSOM in SharePoint Online

In this SharePoint JSOM Example, we will see how to retrieve all list items from SharePoint list using JSOM.

```
<script src="https://ajax.googleapis.com/ajax/libs/jquery/3.3.1/jquery.min.js"></script>
<p style="font-size:20px;" width="500px;" id="Ptask"></p>
<script>
// JavaScript source code
$(document).ready(function () {
ExecuteOrDelayUntilScriptLoaded(retrivetasklistitems, "sp.js");
});
var colltaskListItem;
function retrivetasklistitems() {
var clientContext = new SP.ClientContext.get_current();
var oList = clientContext.get_web().get_lists().getByTitle('TaskList');
var camlQuery = new SP.CamlQuery();
colltaskListItem = oList.getItems(camlQuery);
clientContext.load(colltaskListItem);
clientContext.executeQueryAsync(Function.createDelegate(this,
this.OnQuerySucceeded),
Function.createDelegate(this, this.OnQueryFailed));
}
function OnQuerySucceeded(sender, args) {
var listItemEnumerator = colltaskListItem.getEnumerator();
var tasks = '';
while (listItemEnumerator.moveNext()) {
var oListItem = listItemEnumerator.get_current();
var url =
"https://onlysharepoint2013.sharepoint.com/sites/Raju/tsinfo/Lists/TaskList/DispForm.as
px?ID=" + oListItem.get_item('ID');
tasks += "<a href='" + url + "'>" + oListItem.get_item('Title') + "</a>" + "<br />";
}
```

```
$("#Ptask").html(tasks);
}
function OnQueryFailed(sender, args) {
alert('Error: ' + args.get_message() + '\n' + args.get_stackTrace());
}
</script>
```

Once you run the code, you can see the output will come like below:



Example-12: Retrieve users from SharePoint group using JavaScript

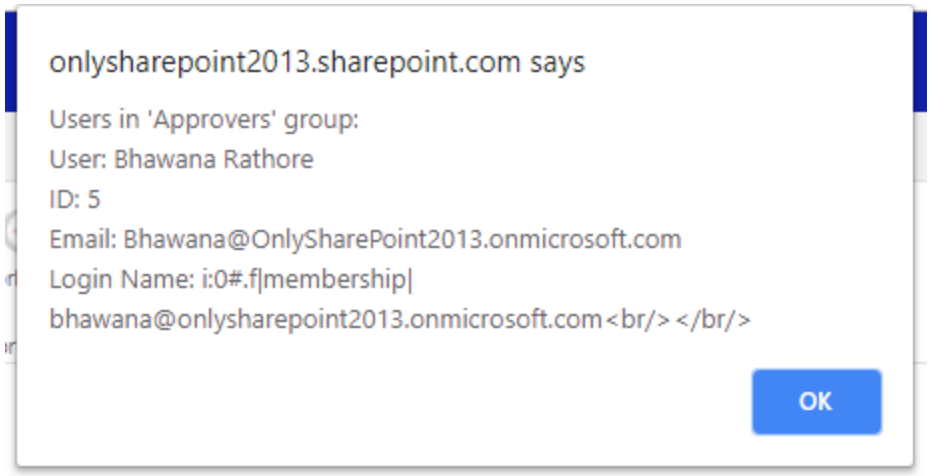
In this SharePoint jsom example, I will show you how to retrieve users from a SharePoint group using the JavaScript object model (jsom) in SharePoint Online or SharePoint 2019/2016/2013.

```
<script src="https://ajax.googleapis.com/ajax/libs/jquery/3.2.1/jquery.min.js"></script>
<script>
$(document).ready(function() {
```

```
SP.SOD.executeFunc('sp.js', 'SP.ClientContext', readUserGrp);
});
function readUserGrp ()
{
var groupName = "Approvers"
//Groups are usually inherited, for this example we grab the root web using the OOTB
spPageContextInfo helper
var ctx = new SP.ClientContext(_spPageContextInfo.siteServerRelativeUrl);
var siteColWeb = ctx.get_web();
//Get groups, then query for specific group by name
var collGroup = siteColWeb.get_siteGroups();
//Use the helper
var oGroup = collGroup.getByName(groupName);
var collUser = oGroup.get_users();
ctx.load(collUser);
ctx.executeQueryAsync(
function(){
console.log("Query success");
var userInfo = '';
var userEnumerator = collUser.getEnumerator();
while (userEnumerator.moveNext()) {
var oUser = userEnumerator.get_current();
userInfo += '\nUser: ' + oUser.get_title() +
'\nID: ' + oUser.get_id() +
'\nEmail: ' + oUser.get_email() +
'\nLogin Name: ' + oUser.get_loginName() + "<br/></br/>";
}
alert("Users in '" + groupName +"' group: " + userInfo);
```

```
},
function(sender, args){ alert('Error: ' + args.get_message()); }
);
}
</script>
```

You can see the output will looks like below:



Example-13: Check user has Delete permission or not using jsom in SharePoint Online

In this example, I will explain how to check the user is having delete permission or not using JavaScript Object Model (jsom) in SharePoint Online.

```
<script src="https://ajax.googleapis.com/ajax/libs/jquery/3.2.1/jquery.min.js"></script>
<script>
$(document).ready(function() {
SP.SOD.executeFunc('sp.js', 'SP.ClientContext', readUserPermission);
});
function readUserPermission ()
```

```
{
var ctx = new SP.ClientContext.get_current();
var curWeb = ctx.get_web();
//In this example we check to see if user can delete list items on the current web.
//Output SP.PermissionKind for all possible permissions. Not all pertain to Web scope!
var basePerm = new SP.BasePermissions();
basePerm.set(SP.PermissionKind.deleteListItems);
var result = curWeb.doesUserHavePermissions(basePerm);
ctx.executeQueryAsync(
function(){ alert(result.get_value() ? "Has permissions" : "Does not have permissions") },
function(sender, args){ alert('Error: ' + args.get_message()); }
);
}
</script>
```

Once you run the above code, it will display in an alert box, the user has delete permission in SharePoint Online.

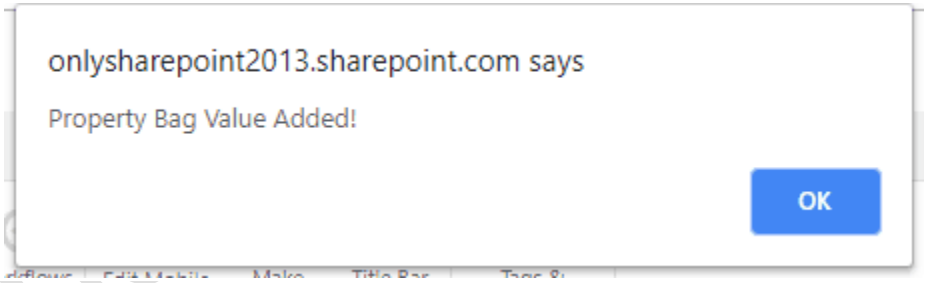
Example-14: Create a Property Bag element using JSOM in SharePoint Online

Here, we will see how to create a Property Bag element using JSOM in SharePoint Online.

```
<script src="https://ajax.googleapis.com/ajax/libs/jquery/3.2.1/jquery.min.js"></script>
<script>
$(document).ready(function() {
SP.SOD.executeFunc('sp.js', 'SP.ClientContext', createPropertyValue);
});
```

```
function createPropertyValue ()
{
var ctx = new SP.ClientContext.get_current();
var curWeb = ctx.get_web();
var propBag = curWeb.get_allProperties();
propBag.set_item("MyProp", "12345");
curWeb.update();
ctx.executeQueryAsync(
function(){ alert('Property Bag Value Added!'); },
function(sender, args){ alert('Error: ' + args.get_message()); }
);
}
</script>
```

Below is the MyProp property bag created with value 12345. You can see a successful message like below:

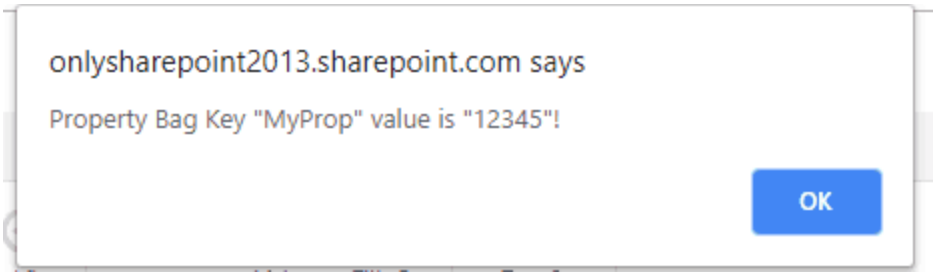


Example-15: Read Property Bag value using JavaScript Object Model (jsom) in SharePoint Online

Now, we will discuss how to read property bag value using JavaScript in SharePoint Online. Here we are fetching a property bag (metadata) MyProp which is returning 12345 using JSOM.

```
<script src="https://ajax.googleapis.com/ajax/libs/jquery/3.2.1/jquery.min.js"></script>
<script>
$(document).ready(function() {
SP.SOD.executeFunc('sp.js', 'SP.ClientContext', readPropertyValue);
});
function readPropertyValue ()
{
var ctx = new SP.ClientContext.get_current();
var curWeb = ctx.get_web();
var propBag = curWeb.get_allProperties();
ctx.load(propBag);
ctx.executeQueryAsync(
function(){ alert('Property Bag Key "MyProp" value is  "' + propBag.get_item("MyProp")
+ '"!'); },
function(sender, args){ alert('Error: ' + args.get_message()); }
);
}
</script>
```

Once you run the above code, MyProp property bag value fetched like below:

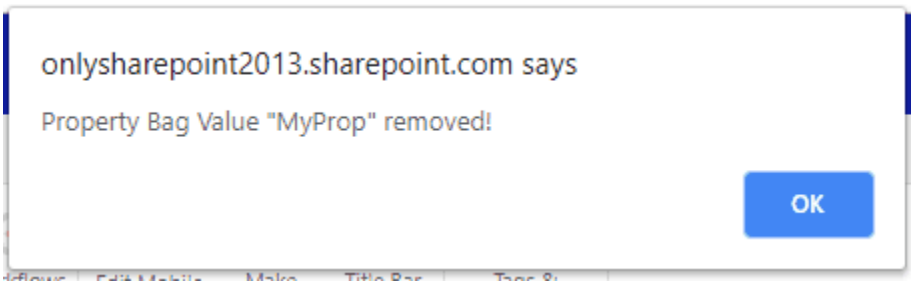


Example-16: Delete property from Property Bag using jsom SharePoint Online

This example explains how to delete property or value from a property bag using jsom (JavaScript Object Model) in SharePoint online. Here we are removing a property bag (metadata) MyProp which we created in the previous example using JSOM.

```
<script src="https://ajax.googleapis.com/ajax/libs/jquery/3.2.1/jquery.min.js"></script>
<script>
$(document).ready(function() {
SP.SOD.executeFunc('sp.js', 'SP.ClientContext', deletePropertyValue);
});
function deletePropertyValue ()
{
var ctx = new SP.ClientContext.get_current();
var curWeb = ctx.get_web();
var propBag = curWeb.get_allProperties();
//setting to null will completely remove key/value from property bag
propBag.set_item("MyProp", null);
curWeb.update();
ctx.executeQueryAsync(
function(){ alert('Property Bag Value "MyProp" removed!'); },
function(sender, args){ alert('Error: ' + args.get_message()); }
);
}
</script>
```

Below is the pop up message after removing MyProp property bag.



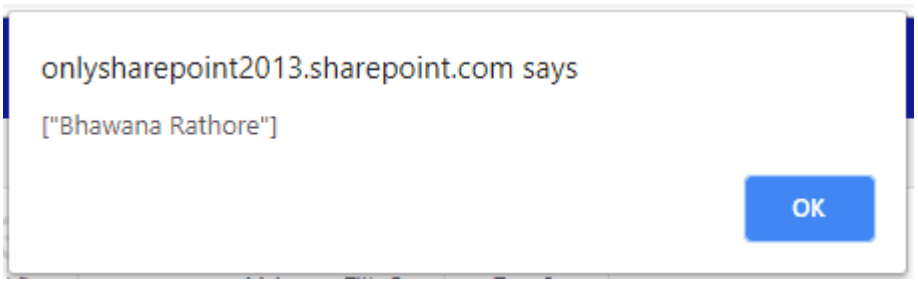
Example-17: Read User Details from Email ID by using JSOM in SharePoint Online/2019/2016/2013

Here we will read user's display name from Email ID using jsom in SharePoint.

```
<script src="https://ajax.googleapis.com/ajax/libs/jquery/3.2.1/jquery.min.js"></script>
<script>
$(document).ready(function() {
SP.SOD.executeFunc('sp.js', 'SP.ClientContext', readUser);
});
function readUser ()
{
var targetUser = "i:0#.f|membership|bhawana@onlysharepoint2013.onmicrosoft.com";
//Global variable
var userProfileProperties;
// Ensure that the SP.UserProfiles.js file is loaded before the custom code runs.
SP.SOD.executeFunc('SP.js', 'SP.ClientContext', function() {
// Make sure PeopleManager is available
SP.SOD.executeFunc('userprofile', 'SP.UserProfiles.PeopleManager', function() {
getUserProperties();
});
});
function getUserProperties() {
```

```
var clientContext = new SP.ClientContext.get_current();
var peopleManager = new SP.UserProfiles.PeopleManager(clientContext);
//Properties from user profiles (These are CASE SENSITIVE)
var profilePropertyNames = ["PreferredName"];
var userProfilePropertiesForUser =
new SP.UserProfiles.UserProfilePropertiesForUser(
clientContext,
targetUser,
profilePropertyNames);
userProfileProperties =
peopleManager.getUserProfilePropertiesFor(userProfilePropertiesForUser);
clientContext.load(userProfilePropertiesForUser);
clientContext.executeQueryAsync(onRequestSuccess, onRequestFail);
}
// This function runs if the executeQueryAsync call succeeds.
function onRequestSuccess() {
var messageText = "\"PreferredName\" property is "
+ userProfileProperties[0];
alert(JSON.stringify(userProfileProperties));
}
// This function runs if the executeQueryAsync call fails.
function onRequestFail(sender, args) {
alert(args.get_message());
}
}
</script>
```

Based on the given user mail id, its fetching users name as shown below.



Example-18: Retrieve Content types using JavaScript object model (jsom) in SharePoint

The below code explains how to retrieve all content types using jsom () in SharePoint 2019/2016/2013 or Online.

```
<script src="https://ajax.googleapis.com/ajax/libs/jquery/3.2.1/jquery.min.js"></script>
<script>
$(document).ready(function() {
SP.SOD.executeFunc('sp.js', 'SP.ClientContext', getContentTypes);
});
function getContentTypes ()
{
var ctx = new SP.ClientContext.get_current();
var web = ctx.get_web();
var cts = web.get_contentTypes();
ctx.load(cts);
ctx.executeQueryAsync(onRequestSuccess, onRequestFail);
// This function runs if the executeQueryAsync call succeeds.
function onRequestSuccess() {
var enumerator = cts.getEnumerator();
while (enumerator.moveNext()) {
var curCt = enumerator.get_current();
alert("CTID: " + curCt.get_id() + " | " + curCt.get_name());
```

```
}
}
// This function runs if the executeQueryAsync call fails.
function onRequestFail(sender, args) {
alert(args.get_message());
}
}
</script>
```

Once you run the above code, it will display all the content type id and corresponding name in SharePoint Online.

Example-19: How to Add Columns or Fields to SharePoint List or Document Library using JSOM

Here we will add fields or columns to a SharePoint custom list using jsom.

We will provide the user an input form to enter column name, data type and description and a submit button.

Once the user fills the details like column name, data type, and description and clicks on Submit, the column will be created or added to the custom list.

HTML File (HTML code for the input controls used to create column):
Here we have taken two text boxes one dropdown list and a submit button. Here I have hard corded the dropdown values as the column types as Single-line, multi-line, and number type.

Have taken a <p> tag to display the success message.

Below is HTML code:

```
<!DOCTYPE HTML>
<HTML>
<head>
<script
src="https://ajax.googleapis.com/ajax/libs/jquery/3.2.1/jquery.min.js"></script></head>
<body>
<table>
<tr>
<td>
Display Name
</td>
<tdcolspan=”4″>
<inputtype=”text”id=”txtcolDisplayNmae”>
</td>
</tr>
<tr>
<td>
columnsType
</td>
<tdcolspan=”4″>
<selectid=”ddlcolumnstype”>
<option>Single line</option>
<option>multi line</option>
<option>number</option>
</select>
</td>
```

```
</tr>
<tr>
<td>
Description
</td>
<tdcolspan=”4″>
<textarearows=”4″cols=”22″id=”txtcolDescr”></textarea>
</td>
</tr>
<tr>
<tdcolspan=”3″>
</td>
<td>
<tdcolspan=”4″>
<inputtype=”button”value=”AddColumns”id=”btncreatecol”></input>
</td>
</tr>
</table>
<p id=”psuccess” align=”center”></p>
</body>
</html>
```

Code – (Creating columns in List from User inputs)

We have taken a custom list and its name as “myDepartmentStore” where the columns will be added.

Here we will take the inputs from the user like type of Column, Display Name and description.

We will retrieve all those inputs and bind using JSOM to create desired columns.

Below is JSOM code to create Columns.

```
$(document).ready(function () {
$("#btncreatecol").click(function(){
retrievecolumnsTemplate();
});
});
functionretrievecolumnsTemplate(){
varcoldisplayname=$("#txtcolDisplayNmae").val();
vartemplate=$('#ddlcolumnstype :selected').val();
varcoldescription=$("#txtcolDescr").val();
varclientContext=newSP.ClientContext.get_current();
varoWebsite=clientContext.get_web();
oList = clientContext.get_web().get_lists().getByTitle('myDepartmentStore');
varfldCollection=oList.get_fields();
if ( template=='multi line'){
varrichTextField=clientContext.castTo(
fldCollection.addFieldAsXml('<Field Type="Note" DisplayName=\"NewField\"
Name="NewField" Required="False" NumLines="12″ RichText="TRUE"
AppendOnly="TRUE" />', true,
SP.AddFieldOptions.addToDefaultContentType),SP.FieldMultiLineText);
richTextField.set_description(coldescription);
richTextField.set_title(coldisplayname);
richTextField.update();
}
```

```
elseif(template==’Single line’ )
{
varsinglelinetext=clientContext.castTo(
fldCollection.addFieldAsXml(‘<Field Type=”Text” DisplayName=\”NewField\”
Name=”NewField” />’,
true,SP.AddFieldOptions.addToDefaultContentType),SP.FieldText);
singlelinetext.set_title(coldisplayname);
singlelinetext.set_description(coldescription);
singlelinetext.update();
}
elseif(template==’number’)
{
varnumberField=clientContext.castTo(
fldCollection.addFieldAsXml(‘<Field Type=”Number” DisplayName=\”NewField\”
Name=”NewField” Required=”False” />’, true,
SP.AddFieldOptions.addToDefaultContentType),SP.FieldNumber) ;
numberField.set_title(coldisplayname);
numberField.set_description(coldescription);
numberField.update();
}
clientContext.executeQueryAsync(Function.createDelegate(this,
this.onQuerySucceeded),
Function.createDelegate(this, this.onQueryFailed));
}
functiononQuerySucceeded(sender, args) {
alert(“column added”);
}
function onQueryFailed(sender, args) {
```

```
alert('Error: '+args.get_message() +'\n'+args.get_stackTrace());
}
```

To verify, Open the custom list and you can see the columns has been added to the list like below:

Example-20: Upload documents to SharePoint document library using JSOM

Now this jsom SharePoint example explains how to upload documents to SharePoint document library using javascript object model in SharePoint Online.

Here I have used html file upload control and a button. User will browse a file and click on the button to upload the file to SharePoint document library.

```
<html>
<head>
<script src="https://ajax.googleapis.com/ajax/libs/jquery/3.2.1/jquery.min.js"></script>
<script>
var fileInput;
$(document)
.ready(function () {
fileInput = $("#getFile");
SP.SOD.executeFunc('sp.js', 'SP.ClientContext', myClickRegister);
});
```

```
function myClickRegister () {
//Register File Upload Click Event
$("#addFileButton").click(readFile);
}
var arrayBuffer;
function readFile() {
//Get File Input Control and read th file name
var element = document.getElementById("getFile");
var file = element.files[0];
var parts = element.value.split("\\");
var fileName = parts[parts.length - 1];
//Read File contents using file reader
var reader = new FileReader();
reader.onload = function (e) {
uploadFile(e.target.result, fileName);
}
reader.onerror = function (e) {
alert(e.target.error);
}
reader.readAsArrayBuffer(file);
}
var attachmentFiles;
function uploadFile(arrayBuffer, fileName) {
//Get Client Context,Web and List object.
var clientContext = new SP.ClientContext();
var oWeb = clientContext.get_web();
var oList = oWeb.get_lists().getByTitle('Documents');
//Convert the file contents into base64 data
```

```
var bytes = new Uint8Array(arrayBuffer);
var i, length, out = '';
for (i = 0, length = bytes.length; i < length; i += 1) {
out += String.fromCharCode(bytes[i]);
}
var base64 = btoa(out);
//Create FileCreationInformation object using the read file data
var createInfo = new SP.FileCreationInformation();
createInfo.set_content(base64);
createInfo.set_url(fileName);
//Add the file to the library
var uploadedDocument = oList.get_rootFolder().get_files().add(createInfo)
//Load client context and execcute the batch
clientContext.load(uploadedDocument);
clientContext.executeQueryAsync(QuerySuccess, QueryFailure);
}
function QuerySuccess() {
console.log('File Uploaded Successfully.');
alert("File Uploaded Successfully.");
}
function QueryFailure(sender, args) {
console.log('Request failed with error message - ' + args.get_message() + ' . Stack
Trace - ' + args.get_stackTrace());
alert("Request failed with error message - " + args.get_message() + " . Stack Trace - " +
args.get_stackTrace());
}
</script>
</head>
```

```
<body>
<input id="getFile" type="file" /><br /> <input id="addFileButton" type="button"
value="Upload" />
</body>
<html>
```

Example-21: Create SharePoint Online List using JSOM

Basically, this example explains, how to create a SharePoint list using JSOM by accepting its name from the user.

Below you will see one form which accepts the name of the list from the user and create a list with that name.

```
<HTML>
<head>
<script src="https://ajax.googleapis.com/ajax/libs/jquery/3.2.1/jquery.min.js"></script>
<script>
$(function() {
btncreate();
});
function btncreate() {
$('#btncreate').on("click", function() {
var listValue = $('#txtlistname').val();
CreateList(listValue);
});
}
function CreateList(listValue) {
```

```
var context = new SP.ClientContext()
var web = context.get_web();
var listcreation = new SP.ListCreationInformation();
listcreation.set_title(listValue);
listcreation.set_templateType(SP.ListTemplateType.genericList)
this.list = web.get_lists().add(listcreation);
context.load(list);
context.executeQueryAsync(Function.createDelegate(this, this.onsuccess),
Function.createDelegate(this, this.onfailed));
}
function onsuccess() {
var results = list.get_title() + 'Create success';
$('#results').html(results);
}
function onfailed(sender, args) {
alert('Creation Failed' + args.get_message() + '\n' + args.get_stackTrace());
}
</script>
</head>
<body>
<div id="form">
<table>
<tr>
<td>
<p>Enter List Name</p>
</td>
<td> <input type="text" id="txtlistname" /> </td>
</tr>
```

```
<tr>
<td> <input type="button" id="btncreate" value="submit" /> </td>
</tr>
</table>
</div>
<div id="results"></div>
</body>
</html>
```

Below is the output where we have created List created list MyStore by name defined by the user.

Example-22: Delete SharePoint Online List using JavaScript Object Model

Basically, this example depicts how to delete a SharePoint list from JSOM by accepting its name from the user.

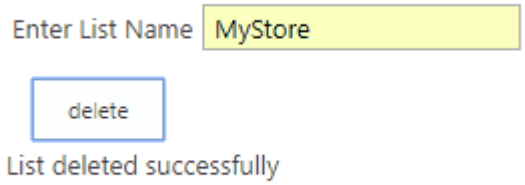
Below you will see one form which accepts the name from the user and removes a list with that name.

```
<HTML>
<head>
<script src="https://ajax.googleapis.com/ajax/libs/jquery/3.2.1/jquery.min.js"></script>
<script>
$(function() {
btndelete()
});
function btndelete() {
$('#btndelete').on("click", function() {
```

```
var listValue = $('#txtlistname').val();
DeleteList(listValue);
});
}
function DeleteList(listValue) {
var context = new SP.ClientContext();
var web = context.get_web();
this.list = web.get_lists().getByTitle(listValue);
list.deleteObject(); // Delete the created list from the site
context.executeQueryAsync(Function.createDelegate(this, this.ondeletesuccess),
Function.createDelegate(this, this.ondeletefailed));
}
function ondeletesuccess() {
$('#results').html("List deleted successfully"); // on success bind the results in HTML
code
}
function ondeletefailed(sender, args) {
alert('Delete Failed' + args.get_message() + '\n' + args.get_stackTrace()); // display the
errot details if deletion failed
} </script>
</head>
<body>
<div id="form">
<table>
<tr>
<td>
<p>Enter List Name</p>
</td>
```

```
<td> <input type="text" id="txtlistname" /> </td>
</tr>  <tr>
<td> <input type="button"     id="btndelete" value="delete" /> </td>
</tr>
</table>
</div>
<div id="results"></div>
</body>
</html>
```

You can see below user will give the list name and click on delete button which will delete the list from the site.



Example-23: Create Site using JSOM in SharePoint Online

In this jsom SharePoint example, we will see how to create Site or Subsite using JSOM (JavaScript Object Model) in SharePoint Online.

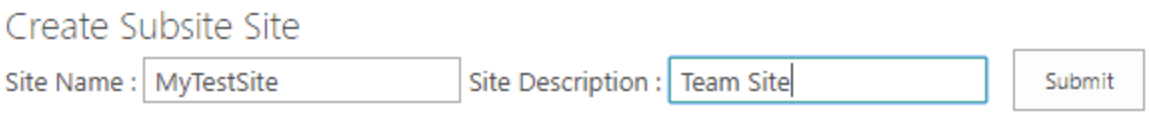
Here we are accepting site name and site description from the user using html form.

```
<html>
<head>
<script src="https://ajax.googleapis.com/ajax/libs/jquery/3.2.1/jquery.min.js"></script>
<script type="text/javascript">
```

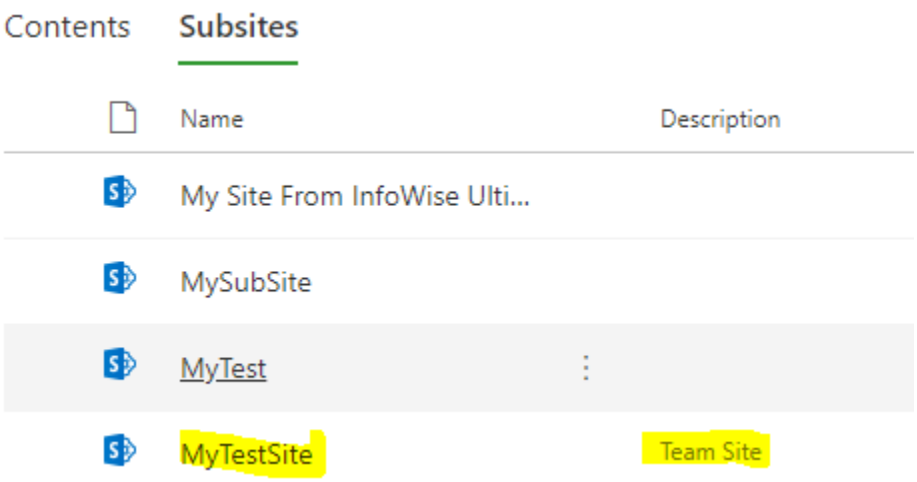
```
$(function() {
bindButtonClick();
});
function bindButtonClick() {
$("#register").on("click", function() {
var name = $("#txtSiteTitle").val();
var id = $("#txtSiteDescription").val();
if (name == '' || id == '') {
alert("Please fill all fields...!!!!!!");
} else {
siteCreation();
}
});
}
function siteCreation () {
var siteTitle = $("#txtSiteTitle").val();
var siteDesc = $("#txtSiteDescription").val();
var siteUrl = siteDesc.replace(/\s/g, "");
var clientContext = new
SP.ClientContext(https://onlysharepoint2013.sharepoint.com/sites/Bhawana/);
var collWeb = clientContext.get_web().get_webs();
var webCreationInfo = new SP.WebCreationInformation();
webCreationInfo.set_title(siteTitle);
webCreationInfo.set_description(siteDesc);
webCreationInfo.set_language(1033);
webCreationInfo.set_url(siteUrl);
webCreationInfo.set_useSamePermissionsAsParentSite(true);
webCreationInfo.set_webTemplate('STS#0');
```

```
var oNewWebsite = collWeb.add(webCreationInfo);
clientContext.executeQueryAsync(Function.createDelegate(this, onQuerySucceeded),
Function.createDelegate(this, this.onQueryFailed));
}
function onQuerySucceeded() {
alert('Site successfully Created!');
}
function onQueryFailed(sender, args) {
alert('Request failed. ' + args.get_message() + '\n' + args.get_stackTrace());
}
</script>
</head>
<body>
<div class="container">
<div class="main">
<form class="form" method="post" action="#">
<h2>Create Subsite Site</h2> <label>Site Name :</label> <input type="text"
name="dname" id="txtSiteTitle"> <label>Site Description :</label> <input type="text"
name="demail" id="txtSiteDescription"> <input type="button" name="register"
id="register" value="Submit"> </form>
</div>
</body>
</html>
```

Once you save the code, it will ask user an input form like below:



Once user click on Submit, it will create the site and you can verify from the Site Contents page.



Example-24: Delete Site using JavaScript Object Model (JSOM) in SharePoint Online

In this jsom SharePoint example, we will see how we can delete a SharePoint site using JavaScript object model (jsom) in SharePoint online.

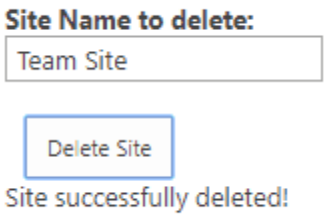
Here we have added a textbox where the user can put the site title in the textbox and the user can click on the submit button which will delete the site and on successful deletion, it will display a successful message to the user.

```
<html>
<head>
```

```
<script src="https://ajax.googleapis.com/ajax/libs/jquery/1.11.2/jquery.min.js"></script>
<script>
$(function () {
bindButtonClick();
});
function bindButtonClick() {
$("#btnSubmit").on("click", function () {
deleteSite();
});
}
function deleteSite() {
var siteTitle = $("#txtSiteTitle").val();
var siteTitleNoSpaces = siteTitle.replace(/\s/g, "");
var siteUrl = _spPageContextInfo.webAbsoluteUrl + "/" + siteTitleNoSpaces;
alert(siteUrl);
var clientContext = new SP.ClientContext(siteUrl);
var oWebsite = clientContext.get_web();
alert(oWebsite);
oWebsite.deleteObject();
clientContext.executeQueryAsync(
Function.createDelegate(this, this.onQuerySucceeded),
Function.createDelegate(this, this.onQueryFailed)
);
}
function onQuerySucceeded() {
$("#divResults").html("Site successfully deleted!");
}
function onQueryFailed(sender, args) {
```

```
alert('Request failed. '+ args.get_message() +
'\n' + args.get_stackTrace());
}
</script></head>
<body>  <div id="DeleteSite">
<div><strong>Site Name to delete:</strong>
<br />
<input type="text" id="txtSiteTitle"/>
</div>
<br />
<input type="button" id="btnSubmit" value="Delete Site" />
</div>
<div id="divResults"></div></body>
</html>
```

You can see it will display a successful message like below:



Example-25: Delete Column from SharePoint list using JSOM (JavaScript Object Model)

In this jsom SharePoint example, we will see how to delete a particular field/column from the list by JSOM.

Here I have my list named MyList and I am deleting one column User Name.

```
<html>
<head>
<meta name="robots" content="noindex, nofollow">
<script src="https://ajax.googleapis.com/ajax/libs/jquery/3.2.1/jquery.min.js"></script>
<script type="text/javascript">
var fieldCollection;
var field;
var list;
function deleteField() {
var clientContext = SP.ClientContext.get_current();
if (clientContext != undefined && clientContext != null) {
var webSite = clientContext.get_web();
this.list = webSite.get_lists().getByTitle("MyList");
this.fieldCollection = list.get_fields();
this.field = fieldCollection.getByTitle("User Name");
this.field.deleteObject();
clientContext.executeQueryAsync(Function.createDelegate(this, this.OnLoadSuccess),
Function.createDelegate(this, this.OnLoadFailed));
}
}
function OnLoadSuccess(sender, args) {
alert("Field deleted successfully.");
}
function OnLoadFailed(sender, args) {
alert('Request failed. ' + args.get_message() + '\n' + args.get_stackTrace());
}
</script>
```

```
<input id="btndeleteField" onclick="deleteField()" type="button" value="Delete the field"
/>
</html>
```

Once you run the above code and click on the button, it will delete the column from the SharePoint list.

Example-26: Create List View in JSOM in SharePoint Online

This SharePoint jsom example, we will see how we can create custom view for SharePoint List using JSOM (JavaScript object model) in SharePoint Online, SharePoint 2019/2016/2013.

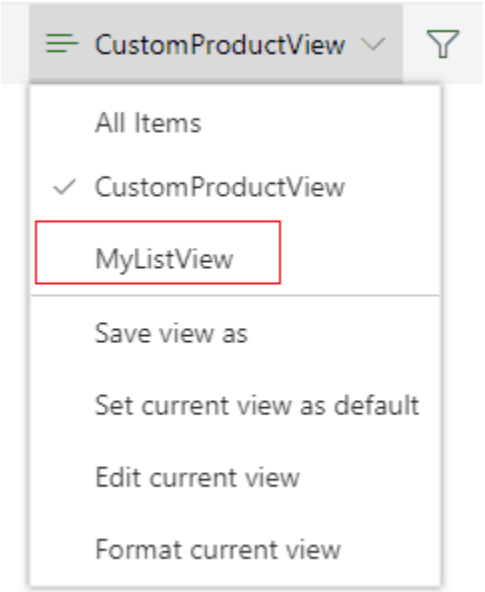
Here I have my list named MyList and I am trying to create my own view with some columns.

```
<script language="javascript" type="text/javascript"
src="//ajax.googleapis.com/ajax/libs/jquery/1.8.1/jquery.min.js"></script>
<script language="javascript" type="text/javascript">
$(document).ready(function() {
SP.SOD.executeFunc('sp.js', 'SP.ClientContext', createListView);
});
var oListViews;
function createListView() {
//Get client context,web and list object
var clientContext = new SP.ClientContext();
var oWebsite = clientContext.get_web();
var oList = oWebsite.get_lists().getByTitle('MyList');
//Set the view fields
```

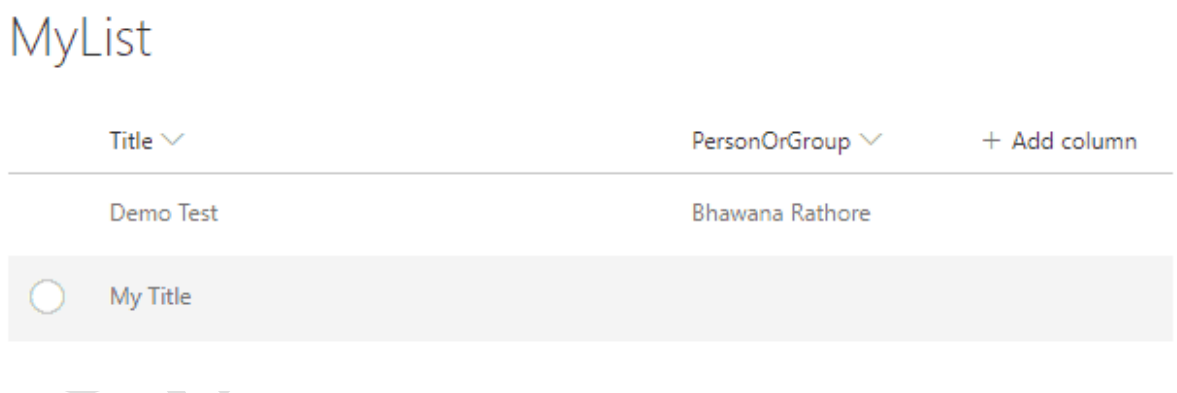
```
var viewFields = new Array('Title','PersonOrGroup');
var viewType = new SP.ViewType();
//Create view using ViewCreationInformation object
var creationInfo = new SP.ViewCreationInformation();
creationInfo.set_title("MyListView");
creationInfo.set_setAsDefaultView("true");
creationInfo.set_rowLimit("10");
creationInfo.set_personalView("false");
creationInfo.set_viewFields(viewFields);
//Set CAML query so that the view shows only a subset of items
var camlQuery = new SP.CamlQuery();
var query = "<Where><IsNotNull><FieldRef Name='Title' /></IsNotNull></Where>";
camlQuery.set_viewXml(query);
creationInfo.set_query(camlQuery);
oListViews = oList.get_views().add(creationInfo);
//Load the client context and execute the batch
clientContext.load(oListViews);
clientContext.executeQueryAsync(QuerySuccess, QueryFailure);
}
function QuerySuccess() {
alert("Views created successfully!");
}
function QueryFailure(sender, args) {
alert ('Request failed' + args.get_message());
}
</script>
```

In the below output, we can see that MyListView has been created for the list MyList with two columns Title and PersonOrGroup.

You can see the view got created like below:



When you open the SharePoint list view, you can see the columns also.



Example-27: Update Listview using JSOM (JavaScript Object Model) in SharePoint Online

In this jsom SharePoint example, we will see how we can update custom view for SharePoint List using JSOM in SharePoint Online or SharePoint 2019/2016/2013.

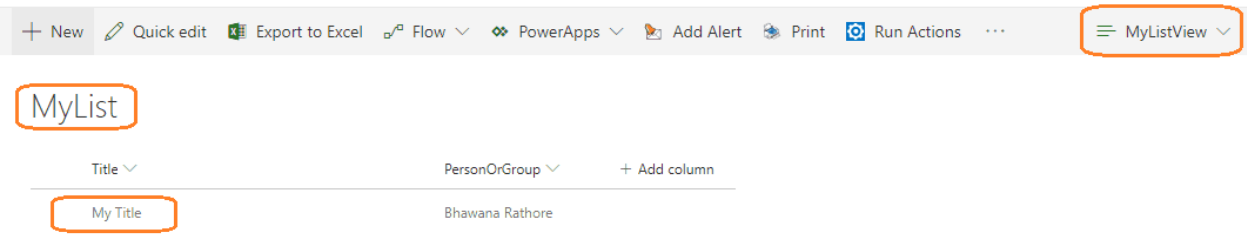
Here I have my list named MyList and I am trying to update view based on particular Title value.

```
<script language="javascript" type="text/javascript"
src="//ajax.googleapis.com/ajax/libs/jquery/1.8.1/jquery.min.js"></script>
<script language="javascript" type="text/javascript">
$(document).ready(function() {
SP.SOD.executeFunc('sp.js', 'SP.ClientContext', UpdateViewJSOM);
});
function UpdateViewJSOM() {
var viewContext = SP.ClientContext.get_current();
var web = viewContext.get_web();
var listCollection = web.get_lists();
list = listCollection.getByTitle('MyList');
viewCollection = list.get_views();
view = viewCollection.getByTitle('MyListView');
var camlQuery = new SP.CamlQuery();
var query = "<Where><Eq>><FieldRef Name='Title' /><Value Type='Text'>My
Title</Value></Eq></Where>";
camlQuery.set_viewXml(query);
view.set_viewQuery(camlQuery);
view.update();
viewContext.load(view);
viewContext.executeQueryAsync(QuerySuccess, QueryFailure);
}
function QuerySuccess() {
alert("Views updated successfully!");
}
```

```
function QueryFailure(sender, args) {
alert ('Request failed' + args.get_message());
}
</script>
```

In below output we can see that MyListView has been updated for the list MyList with two columns Title and PersonOrGroup.

Here we are displaying only entry where Title is MyTitle.



Example-28: Delete List View using JSOM (JavaScript Object Model) in SharePoint Online/2019/2016/2013

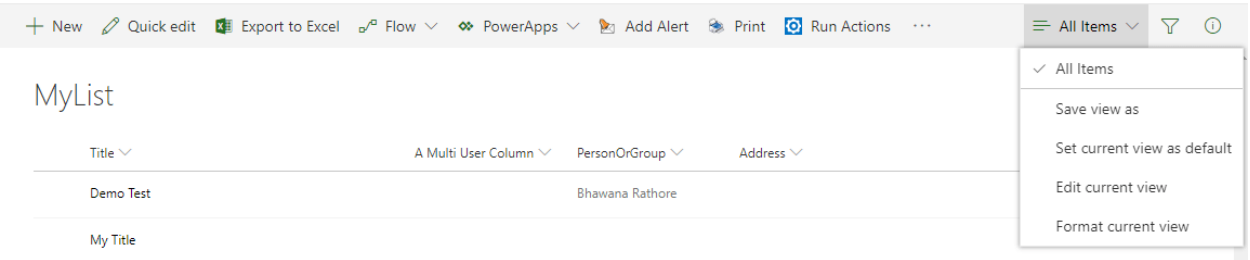
In this jsom SharePoint example, we will see how we can create custom view for SharePoint List using JSOM (JavaScript object model) in SharePoint Online or SharePoint 2019/2016/2013.

Here I have my list named MyList and I am trying to delete my own view.

```
<script language="javascript" type="text/javascript"
src="//ajax.googleapis.com/ajax/libs/jquery/1.8.1/jquery.min.js"></script>
<script language="javascript" type="text/javascript">
$(document).ready(function() {
SP.SOD.executeFunc('sp.js', 'SP.ClientContext', deleteListView);
```

```
});
function deleteListView() {
//Get the client context,web and list object
var clientContext = new SP.ClientContext();
var oWeb = clientContext.get_web();
var oList = oWeb.get_lists().getByTitle('MyList');
//Get the view object to delete
var oView = oList.get_views().getByTitle('MyListView');
oView.deleteObject();
//Execute the batch
clientContext.executeQueryAsync(QuerySuccess, QueryFailure);
}
function QuerySuccess() {
alert('Specified View has been deleted.');
}
function QueryFailure(sender, args) {
alert('Request failed' + args.get_message());
}
</script>
```

In below output we can see that MyListView has been removed from the list MyList in SharePoint Online. You can see, the list view is not available here.



Example-29: How to Get Current user has Edit permission or not using JSOM in SharePoint Online

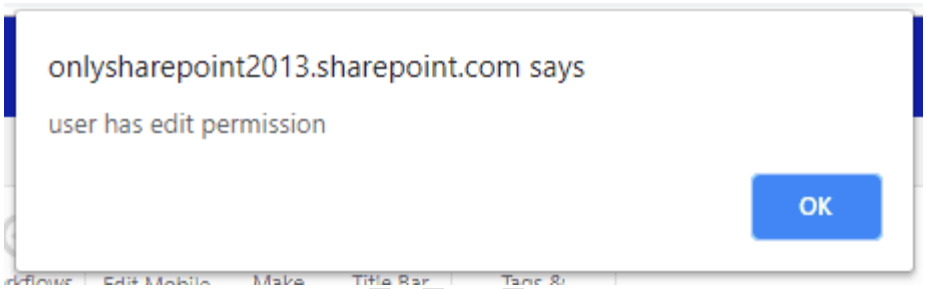
In this jsom SharePoint example, we will see how we can check if the current user has edit permission using JSOM (JavaScript object model) in SharePoint 2019/2016/2013 or SharePoint Online.

Here I have my list named MyList.

```
<script language="javascript" type="text/javascript"
src="//ajax.googleapis.com/ajax/libs/jquery/1.8.1/jquery.min.js"></script>
<script language="javascript" type="text/javascript">
$(document).ready(function() {
SP.SOD.executeFunc('sp.js', 'SP.ClientContext', getCurrentUserPermission);
});
function getCurrentUserPermission()
{
var web,clientContext,currentUser,oList,perMask;
clientContext = new SP.ClientContext.get_current();
web = clientContext.get_web();
currentUser = web.get_currentUser();
oList = web.get_lists().getByTitle('MyList');
clientContext.load(oList,'EffectiveBasePermissions');
clientContext.load(currentUser);
clientContext.load(web);
clientContext.executeQueryAsync(function(){
if (oList.get_effectiveBasePermissions().has(SP.PermissionKind.editListItems)){
alert("user has edit permission");
}else{
```

```
alert ("user doesn't have edit permission");
}
}, function(sender, args){
alert ('request failed ' + args.get_message() + '\n'+ args.get_stackTrace());
});
}
</script>
```

Below is the output where we can see that current user have the edit permission.



Example-30: Get Current Logged in user information using jsom (JavaScript object model) in SharePoint

In this jsom SharePoint example, we will see how to fetch current user information like Name, Id, Email, Title using JSOM in SharePoint Online.

```
<script type="text/javascript">
ExecuteOrDelayUntilScriptLoaded(init,'sp.js');
var currentUser;
function init(){
this.clientContext = new SP.ClientContext.get_current();
this.oWeb = clientContext.get_web();
currentUser = this.oWeb.get_currentUser();
this.clientContext.load(currentUser);
```

```
this.clientContext.executeQueryAsync(Function.createDelegate(this,this.onQuerySucce
eded), Function.createDelegate(this,this.onQueryFailed));
}
function onQuerySucceeded() {
document.getElementById('userLoginName').innerHTML =
currentUser.get_loginName();
document.getElementById('userId').innerHTML = currentUser.get_id();
document.getElementById('userTitle').innerHTML = currentUser.get_title();
document.getElementById('userEmail').innerHTML = currentUser.get_email();
}
function onQueryFailed(sender, args) {
alert('Request failed. \nError: ' + args.get_message() + '\nStackTrace: ' +
args.get_stackTrace());
}
</script>
<div>Current Logged User Info <br>
Login Name :-  <span id="userLoginName"></span></br>
User ID :- <span id="userId"></span></br>
User Title :-<span id="userTitle"></span></br>
Email :-<span id="userEmail"></span></br>
</div>
```

Below is the output where we can see complete information about the current user in SharePoint.

Current Logged User Info
Login Name :- i:0#.f|membership|bhawana@onlysharepoint2013.onmicrosoft.com
User ID :- 5
User Title :-Bhawana Rathore
Email :-Bhawana@OnlySharePoint2013.onmicrosoft.com

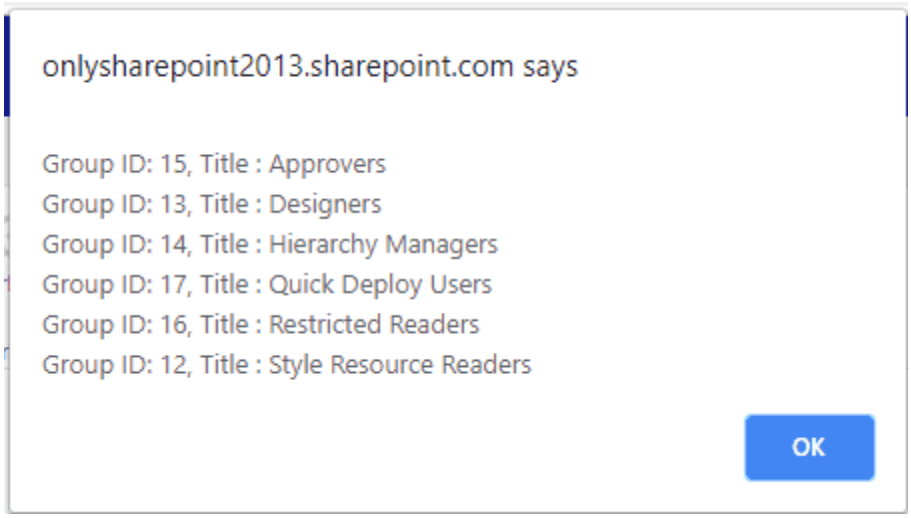
Example-31: Get Group Information of Current Logged In user in SharePoint using JavaScript

Here in this SharePoint 2013 jsom example, we will see how to get group information of current logged in user. This will display all the group names the current user belongs to.

```
<script type="text/javascript">
ExecuteOrDelayUntilScriptLoaded(injectMethod,'sp.js');
var currentUser;
function injectMethod(){
getcurrentuserDetails();
}
function getcurrentuserDetails () {
var clientContext = SP.ClientContext.get_current();
oUser = clientContext.get_web().get_currentUser();
oGroups = oUser.get_groups();
clientContext.load(oUser);
clientContext.load(oGroups);
clientContext.executeQueryAsync(
Function.createDelegate(this, function() {
var groupsInfo = '';
var groupsEnumerator = oGroups.getEnumerator();
console.log('Count of current user groups: ' + oGroups.get_count());
```

```
while (groupsEnumerator.moveNext()) {
var oGroup = groupsEnumerator.get_current();
groupsInfo += '\n' + 'Group ID: ' + oGroup.get_id() + ', ' + 'Title : ' + oGroup.get_title();
}
alert(groupsInfo.toString());
}),
Function.createDelegate(this, function() {
alert('failed');
}));
}
</script>
```

Below is the final output where we can see how many groups does user belongs in SharePoint.



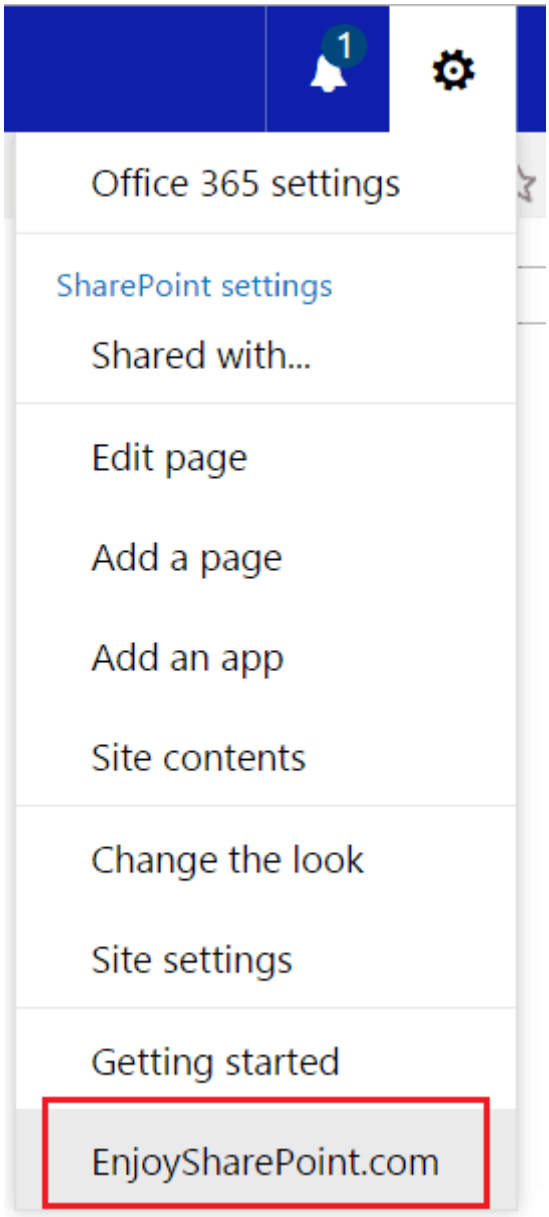
Example-32: Add menu in Site Actions menu in SharePoint Online using JSOM

Here in this jsom SharePoint example, I will display how to add menu items in Site Actions in SharePoint Online using JavaScript object model (jsom).

```
<script src="https://ajax.googleapis.com/ajax/libs/jquery/1.4.2/jquery.min.js"></script>
<input name="ADD" id="btnAdd" type="submit" value="Add new custom actions" />
<script language="javascript" type="text/javascript">
$(document).ready(function () {
$("#btnAdd").click(function () {
AddCustomActions();
});
});
function AddCustomActions() {
var clientContext = new SP.ClientContext();
var web=clientContext.get_web();
var customActions = web.get_userCustomActions();
var newCustomActionMenu = customActions.add();
newCustomActionMenu.set_location('Microsoft.SharePoint.StandardMenu');
newCustomActionMenu.set_group('SiteActions');
newCustomActionMenu.set_sequence(3000);
newCustomActionMenu.set_title('EnjoySharePoint.com');
newCustomActionMenu.set_description('Go to EnjoySharePoint.com');
newCustomActionMenu.set_url('https://www.enjoysharepoint.com');
newCustomActionMenu.update();
clientContext.executeQueryAsync(Function.createDelegate(this, this.onSuccess),
Function.createDelegate(this, this.onFailure));
}
function onSuccess() {
alert('New Item Added to Site Actions Menu');
}
function onFailure(sender, args) {
alert('Error Occurred. ' + args.get_message());
```

```
}
</script>
```

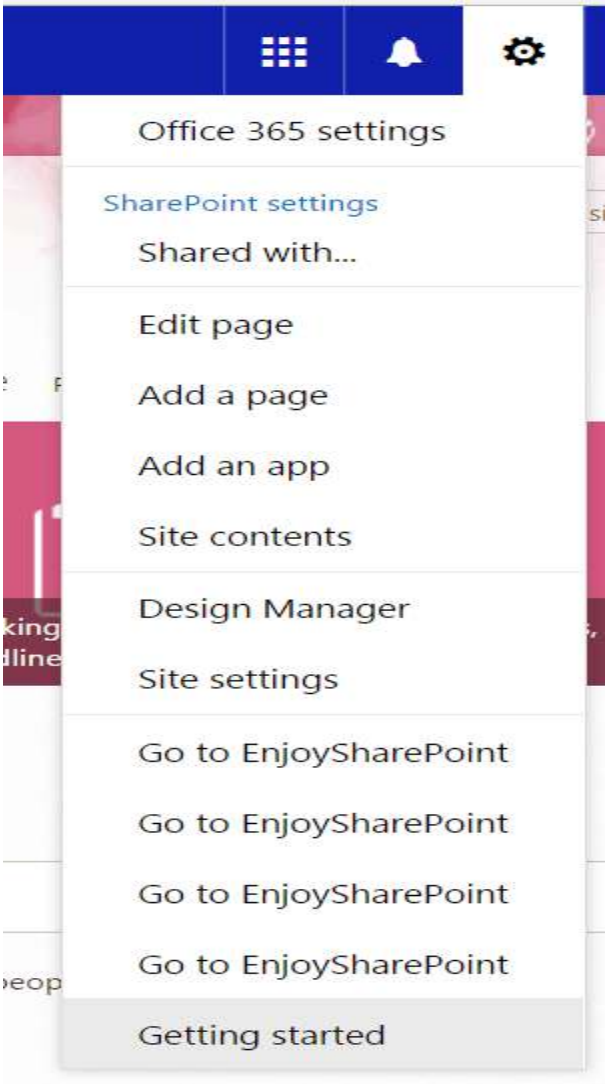
Once you add the code and execute, the menu items will be created as shown below from the Site Actions menu.



Example-33: Remove items from Site Actions menu using JavaScript object model in SharePoint Online Office 365

In this jsom SharePoint tutorial, we will see how to remove items from the Site Actions menu using JavaScript object model (jsom) in SharePoint Online Office 365

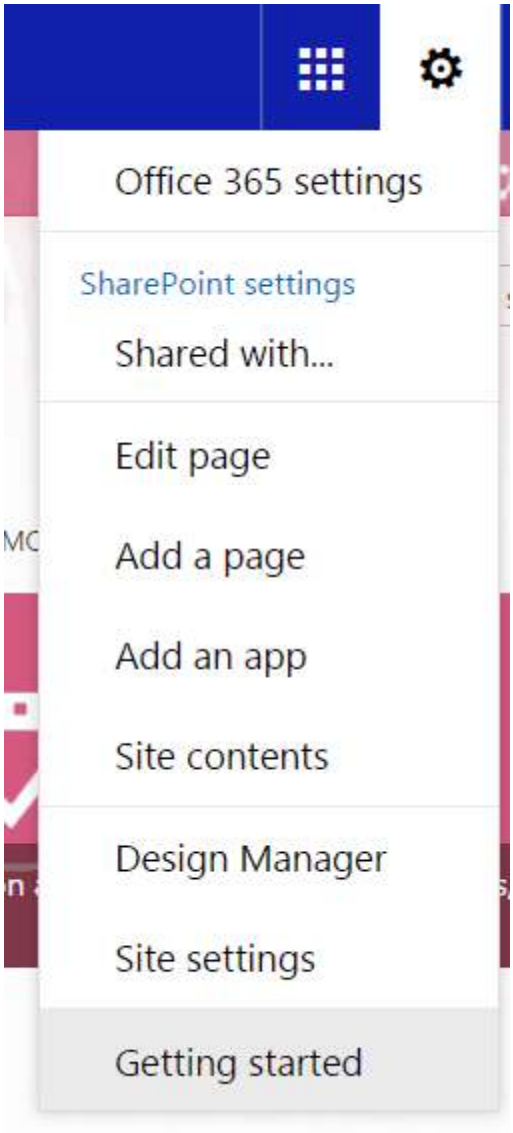
Here in my SharePoint online site, I have added few items to the Site Actions menu like below:



```
<script language=”javascript” type=”text/javascript”
src=”https://ajax.googleapis.com/ajax/libs/jquery/1.8.1/jquery.min.js”></script>
<script language=”javascript” type=”text/javascript”>
var customActions;
var clientContext;
$(document).ready(function() {
SP.SOD.executeFunc(‘sp.js’, ‘SP.ClientContext’, DeleteCustomMenuAction);
});
function DeleteCustomMenuAction() {
clientContext = new SP.ClientContext();
var web = clientContext.get_web();
customActions = web.get_userCustomActions();
clientContext.load(customActions);
clientContext.load(web);
clientContext.executeQueryAsync(OnSuccess, OnFailure);
}
function OnSuccess() {
var enumerator = customActions.getEnumerator();
var removeThese = []
while (enumerator.moveNext()) {
var action = enumerator.get_current();
alert(action.get_title());
if (action.get_title() == “Go to EnjoySharePoint”)
{
action.deleteObject();
}
clientContext.load(action);
clientContext.executeQueryAsync();
```

```
}
}
function OnFailure() {
alert('Fail');
}
</script>
```

Once you add the code and execute, the menu items will be removed from the Site Actions menu.



Example-34: Retrieve site template used in SharePoint Online site using jsom (JavaScript object model)

Here in this jsom SharePoint example, I will show you how to retrieve the site template used in the SharePoint Online site in jsom.

```
<script src="https://ajax.googleapis.com/ajax/libs/jquery/3.3.1/jquery.min.js"></script>
Site Templates used in<b> "https://onlysharepoint2013.sharepoint.com/sites/Bhawana"
</b>site
<p id="ptemp"></p>
<script>
ExecuteOrDelayUntilScriptLoaded(retrieveSiteTemplate, 'sp.js');
var site;
function retrieveSiteTemplate() {
var clientContext = new SP.ClientContext.get_current();
site = clientContext.get_web();
clientContext.load(site);
clientContext.executeQueryAsync(success, failure);
}
function success() {
$("#ptemp").html("Site Template Used:: "+ site.get_webTemplate());
}
function failure() {
alert("Failure!");
}
</script>
```

You can see, if you run the above code, it will display the site template using in SharePoint Online using jsom.

Site Templates used in **"https://onlysharepoint2013.sharepoint.com/sites/Bhawana"** site

Site Template Used:: STS

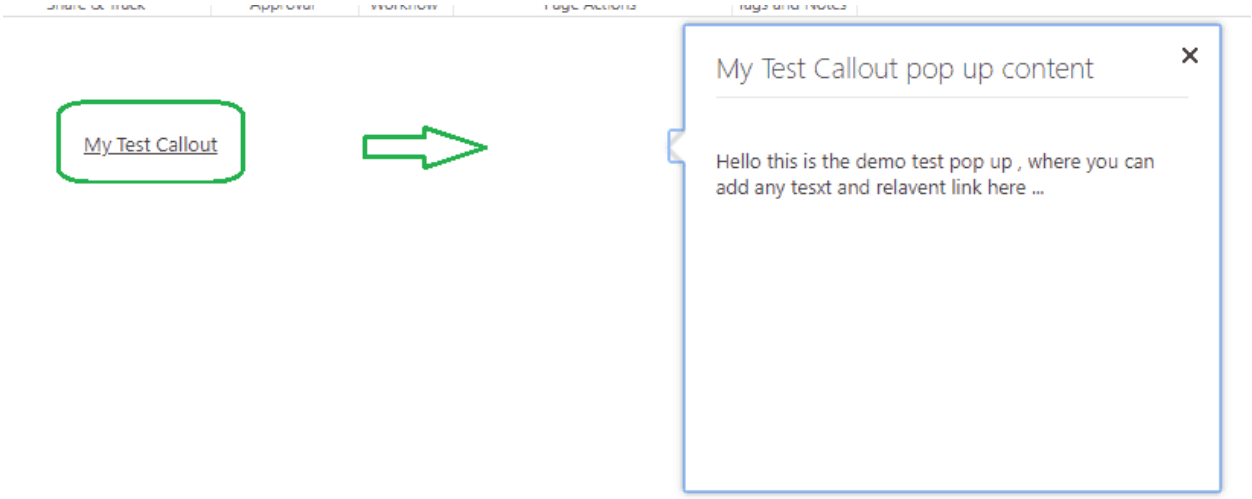
Example-35: How to create a custom callouts tooltip using JavaScript Object Model (jsom) in SharePoint Online

In this jsom SharePoint example, we will see how to create a popup dialog box on click event by using JSOM.

```
<script type="text/javascript">
SP.SOD.executeFunc("callout.js", "Callout", function()
{
var _link = document.getElementById("AboutusLink");
var listCallout = CalloutManager.createNew(
{
launchPoint: _link,
beakOrientation: "leftRight",
ID: "CallOut ID",
title: "My Test Callout pop up content",
content: "<div class=\"ms-soften\" style=\"margin-top:2px; \"><hr/></div>" + "<div
id='confirmationBLOCK' style=\"margin-top:13px;visibility:hidden;\">Thank you for
Contacting Us!</div>" + "<div class=\"callout-section\" style=\"margin-
top:2px;width:95%;Height:200px; \"><span id='CommentsArea'
style=\"width:100%;height: 100%; -webkit-box-sizing: border-box; -moz-box-sizing:
border-box; box-sizing: border-box;\">Hello this is the demo test pop up , where you can
add any tesxt and relavent link here ...</span ></div>",
});
});
</script>
```

```
<div id="AboutusLink" style="width:38%;"><u><span class=\"ms-commandLink\"
style=\"text-align: left;font-size: 14px;\">My Test Callout </span></u></div>
```

Once you run the above code, you can see the callout pop up will appear like below:



Example-36: How to Add user other than logged-in user to SharePoint group using JavaScript

In this SharePoint JavaScript examples, we will discuss how to add user other than logged-in user to SharePoint group using JavaScript object model (JSOM).

Here is the below code, you can just pass in the form of “domain\\username” and also you can pass the group id.

```
<input type='button' value='Add User to SharePoint Group' onclick="clickMethod();"/>
<br />
<script language="javascript" type="text/javascript">
var user;
var spGroup;
function clickMethod() {
```

```
var clientContext = new SP.ClientContext.get_current();
var web=clientContext.get_web();
var siteGroups = clientContext.get_web().get_siteGroups();
spGroup=siteGroups.getById(2544);
//user=clientContext.get_web().get_currentUser();
user=web.ensureUser("domain\\username");
var userCollection=spGroup.get_users();
userCollection.addUser(user);
clientContext.load(user);
clientContext.load(spGroup);
clientContext.executeQueryAsync(onQuerySucceeded, onQueryFailed);
}
function onQuerySucceeded() {
alert('success');
}
function onQueryFailed() {
alert('Request failed.');
}
</script>
```

Once you will run the code, the user will be added to group whose id is 2544 in SharePoint online.

Example-37: Remove user other than logged-in user to SharePoint group using JavaScript

In this SharePoint JavaScript examples, we will see if we want to remove a user from a SharePoint group then we can use the remove() method which usually takes a user as the parameter.

```
<input type='button' value='Remove User from SharePoint Group'
onclick="RemoveUser();"/>
<br />
<script language="javascript" type="text/javascript">
var user;
var spGroup;
function RemoveUser() {
var clientContext = new SP.ClientContext.get_current();
var web=clientContext.get_web();
var siteGroups = clientContext.get_web().get_siteGroups();
spGroup=siteGroups.getById(2544);
//user=clientContext.get_web().get_currentUser();
user=web.ensureUser("DomainName\\username");
var userCollection=spGroup.get_users();
userCollection.remove(user);
clientContext.executeQueryAsync(onQuerySucceeded, onQueryFailed);
}
function onQuerySucceeded() {
alert('success');
}
function onQueryFailed() {
alert('Request failed.');
}
</script>
```

Once you run the code, it will remove the user from the SharePoint group in SharePoint.

Example-38: Add Site Column to Content Type using JavaScript Object Model (jsom)

In this jsom SharePoint examples, we will see how to add a site column to content type using the JavaScript object model in SharePoint Online or SharePoint 2013/2016/2019.

```
<script language="javascript" type="text/javascript"
src="//ajax.googleapis.com/ajax/libs/jquery/1.8.1/jquery.min.js"></script>
<script language="javascript" type="text/javascript">
$(document).ready(function() {
SP.SOD.executeFunc('sp.js', 'SP.ClientContext', AddContentType);
});
function AddContentType() {
var ctx = new SP.ClientContext.get_current();
var field = ctx.get_site().get_rootWeb().get_fields()
.getByInternalNameOrTitle("Company_x0020_Branches");
var ctype = ctx.get_site().get_rootWeb().get_contentTypes()
.getById("0x0100BD927BBDD704954CB6FEA329CB809491");
ctx.load(ctype);
ctx.load(field);
var createInfo = new SP.FieldLinkCreationInformation();
createInfo.set_field(field);
var fieldLink = ctype.get_fieldLinks().add(createInfo);
ctype.update (true);
ctx.load(fieldLink);
ctx.executeQueryAsync(success, failure);
}
function success() {
alert('success');
```

```
}
function failure() {
alert('failure');
}
</script>
<button type="button" onclick="AddContentType();">Add Site Column to Content
Type</button>
```

Once you run the code and user click on the button, it will add the site column to the content type in SharePoint Online, SharePoint 2019/2016/2013.

Example-39: How to delete file from document library using JavaScript object model (jsom) in SharePoint Online

In this SharePoint JavaScript example, we will see how to delete a file from a SharePoint document library using JavaScript object model (JSOM).

```
<html>
<head>
<script src="https://ajax.googleapis.com/ajax/libs/jquery/1.11.2/jquery.min.js"></script>
<script>
$(function () {
bindButtonClick();
});
function bindButtonClick() {
$("#btnSubmit").on("click", function () {
deleteDocument();
});
}
```

```
function deleteDocument() {
var docTitle = $("#txtDocumentTitle").val() + ".txt";
var clientContext = new SP.ClientContext();
var oWebsite = clientContext.get_web();
var fileUrl = _spPageContextInfo.webServerRelativeUrl + "/Shared Documents/" +
docTitle;
this.fileToDelete = oWebsite.getFileByServerRelativeUrl(fileUrl);
this.fileToDelete.deleteObject();
clientContext.executeQueryAsync(
Function.createDelegate(this, this.onQuerySucceeded),
Function.createDelegate(this, this.onQueryFailed)
);
}
function onQuerySucceeded() {
$("#divResults").html("Document successfully deleted!");
}
function onQueryFailed(sender, args) {
alert('Request failed. ' + args.get_message() +
'\n' + args.get_stackTrace());
}
</script>
</head>
<body>
<div id="DeleteFile">
<div>
<strong>Enter File Name to Delete:</strong>
<br />
<input type="text" id="txtDocumentTitle" />
```

```
</div>
<br />
<input type="button" id="btnSubmit" value="Delete File" />
</div>
<div id="divResults"></div>
</body>
</html>
```

Once you run the code, user will input the file name and click on button, the file will be deleted in SharePoint 2013/2016/2019 or SharePoint Online.

Example-40: Create a folder inside document library using the JavaScript object model (jsom) in SharePoint Online Office 365

In this SharePoint JavaScript examples, we will see how to create a folder inside the document library using the JavaScript object model (jsom) in SharePoint Online Office 365.

HTML Code:

Below is the full HTML code where we have to take an HTML textbox and an HTML button.

```
<div>
<strong>Enter FolderName:</strong><br />
<input type="text" id="txtFolderName" /><br />
<input type="button" id="btnSubmit" value="Create Folder" />
</div>
<div id="divResults"></div>
```

JSOM Code:

Below is the full jsom code. Here also we have given a reference to the jQuery min file because we have used jQuery to bind the button click even as well as we are also retrieving the textbox value using jQuery like below:

```
var folderName = $("#txtFolderName").val();
```

Here we are creating the folder inside the "Documents" document library, if you want to give any other document library then you can just replace the document library name is the below line.

```
var oList = oWebsite.get_lists().getByTitle("Documents");
```

Below is the full jsom code:

```
<script src="https://ajax.googleapis.com/ajax/libs/jquery/1.11.2/jquery.min.js"></script>
<script>
$(function () {
bindButtonClick();
});
function bindButtonClick() {
$("#btnSubmit").on("click", function () {
createFolder();
});
}
function createFolder() {
var folderName = $("#txtFolderName").val();
var clientContext = new SP.ClientContext();
```

```
var oWebsite = clientContext.get_web();
var oList = oWebsite.get_lists().getByTitle("Documents");
var folderCreateInfo = new SP.ListItemCreationInformation();
folderCreateInfo.set_underlyingObjectType(SP.FileSystemObjectType.folder);
folderCreateInfo.set_leafName(folderName);
this.oListItem = oList.addItem(folderCreateInfo);
this.oListItem.update();
clientContext.load(oList);
clientContext.executeQueryAsync(
Function.createDelegate(this, this.onQuerySucceeded),
Function.createDelegate(this, this.onQueryFailed)
);
}
function onQuerySucceeded() {
$("#divResults").html("Folder successfully created!");
}
function onQueryFailed(sender, args) {
alert('Request failed. ' + args.get_message() +
'\n' + args.get_stackTrace());
}
</script>
```

Once you run the above code, the user will enter a folder name and click on the button, which will create a folder inside a document library in SharePoint Online.

Example-41: Delete Folder inside Document Library in JavaScript object model (jsom) in SharePoint Online

This jsom SharePoint example explains, how to delete folder inside the document library in SharePoint Online using jsom (JavaScript Object Model).

```
<input type='button' id='id1' value='Delete Folder using JSOM'
onclick="DeleteFolderJSOM();"/>
<script type="text/javascript">
function DeleteFolderJSOM()
{
var webUrl = _spPageContextInfo.webAbsoluteUrl;
var context = new SP.ClientContext.get_current();
var website = context.get_web();
var folderUrl = webUrl + "/Shared Documents/Folder1";
var folderToDelete = website.getFolderByServerRelativeUrl(folderUrl);
folderToDelete.deleteObject();
context.executeQueryAsync(success,failure);
}
function success()
{
alert('Folder deleted successfully');
}
```

```
function failure()
{
alert('Error while deleting folder');
}
</script>
```

The above code will delete folder from SharePoint document library using jsom.

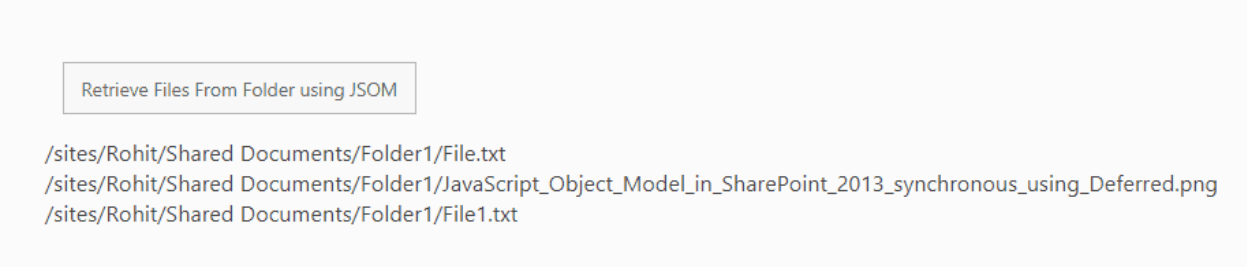
Example-42: Retrieve Files from Folder in SharePoint document library using jsom

In this SharePoint JavaScript examples, we will discuss how to read all file content using the JavaScript object model (jsom) in SharePoint Online Office 365.

```
<input type='button' id='id1' value='Retrieve Files From Folder using JSOM'
onclick="ShowAllFilesFromFolder();"/>
<p id="demo"></p>
<script type="text/javascript">
var files;
function ShowAllFilesFromFolder()
{
var webUrl = _spPageContextInfo.webAbsoluteUrl;
var context = new SP.ClientContext.get_current();
var web = context.get_web();
var folderUrl = webUrl + "/Shared Documents/Folder1";
var folder = web.getFolderByServerRelativeUrl(folderUrl);
files = folder.get_files();
context.load(files);
context.executeQueryAsync(success,failure);
```

```
}
function success()
{
var docURLs = '';
var listItemEnumerator = files.getEnumerator();
while (listItemEnumerator.moveNext()) {
var fileUrl = listItemEnumerator.get_current().get_serverRelativeUrl();
docURLs +=fileUrl+"<br>";
}
document.getElementById("demo").innerHTML = docURLs;
}
function failure()
{
alert('Error while deleting folder');
}
</script>
```

Once you run the above jsom code, you can see it will display file paths in SharePoint document library.



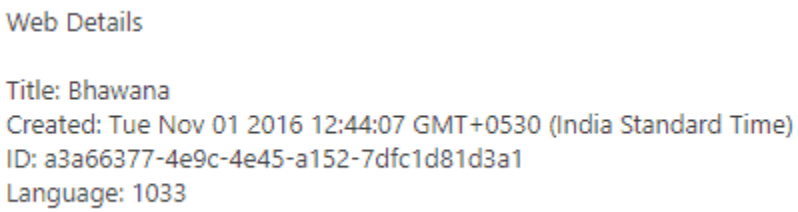
Example-43: Retrieve SharePoint web properties using jsom (JavaScript Object Model)

In this SharePoint JavaScript examples, we will discuss how to read website properties using JSOM (JavaScript object model) in SharePoint Online or SharePoint 2019/2016/2013.

```
<script language="javascript" type="text/javascript"
src="//ajax.googleapis.com/ajax/libs/jquery/1.8.1/jquery.min.js"></script>
<script language="javascript" type="text/javascript">
$(document).ready(function() {
SP.SOD.executeFunc('sp.js', 'SP.ClientContext', readWebProperties);
});
function readWebProperties ()
{
var context = SP.ClientContext.get_current();
var web = context.get_web();
var webDetails="";
context.load(web);
context.load(web);
context.executeQueryAsync(function()
{
webDetails += "Web Details <br/>";
webDetails += "<br/> Title: " + web.get_title();
webDetails += "<br/> Created: " + web.get_created();
webDetails += "<br/> ID: " + web.get_id();
webDetails += "<br/> Language: " + web.get_language();
var insert = document.getElementById('webDetails');
insert.innerHTML = webDetails;
```

```
},
function(sender, args)
{
alert('Failed to get the web Details. Error:' + args.get_message());
});
}
</script>
<div id="webDetails">
</div>
```

Below is the output where we can see SharePoint web site details like Name, Created, language.



Example-44: Retrieve web properties using JSOM in SharePoint Online/2019/2016/2013

Here we will discuss how to write to web site property using JSOM (JavaScript object model) in SharePoint Online.

```
<script language="javascript" type="text/javascript"
src="//ajax.googleapis.com/ajax/libs/jquery/1.8.1/jquery.min.js"></script>
<script language="javascript" type="text/javascript">
$(document).ready(function() {
SP.SOD.executeFunc('sp.js', 'SP.ClientContext', readWebProperties);
```

```
});
function readWebProperties ()
{
var context = SP.ClientContext.get_current();
var web = context.get_web();
var webDetails="";
context.load(web);
web.set_description("Setting the web demo description through Jsom");
context.load(web);
context.executeQueryAsync(function()
{
webDetails += "Web Details <br/>";
webDetails += "<br/> Title: " + web.get_title();
webDetails += "<br/> Created: " + web.get_created();
webDetails += "<br/> Description: " + web.get_description();
webDetails += "<br/> ID: " + web.get_id();
webDetails += "<br/> Language: " + web.get_language();
var insert = document.getElementById('webDetails');
insert.innerHTML = webDetails;
},
function(sender, args)
{
alert('Failed to get the web Details. Error:' + args.get_message());
});
}
</script>
<div id="webDetails">
</div>
```

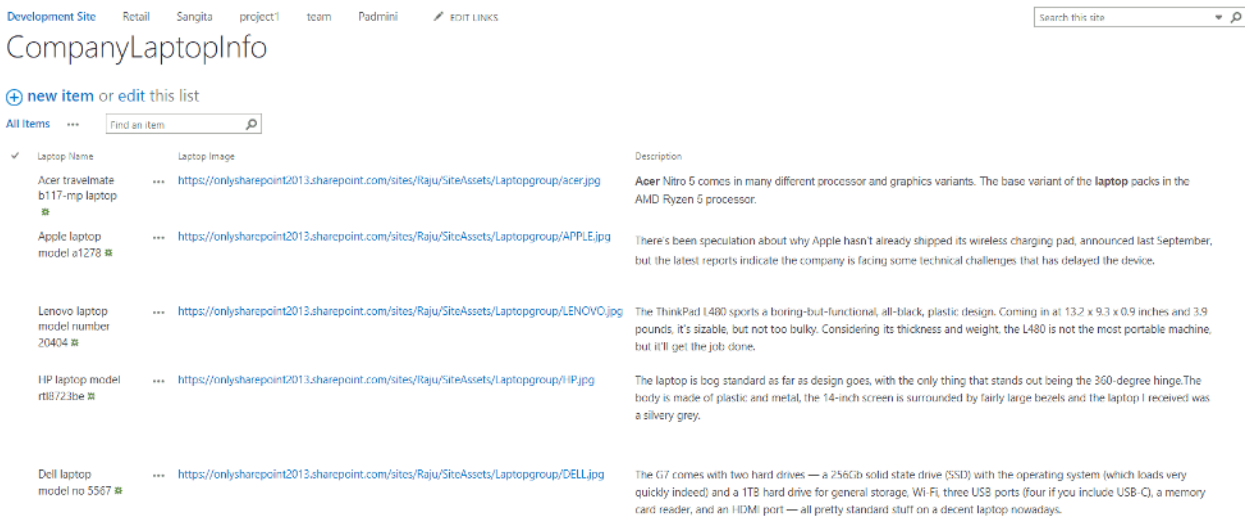
Below is the output where we can see web site updated Description.

Example-45: How to Display List Items in Div using JSOM (JavaScript Object Model)

In this SharePoint JavaScript example, I have a SharePoint list, and we will display list items in a div using JavaScript in SharePoint Online.

- Laptop Name (“Single line of text”)
- Laptop Image (“Hyperlink” which is having the image URL)
- Description (“Multi-line of text”)

The list looks like below:



HTML Code

```
<!DOCTYPE html>
```

```
<html lang="en" xmlns="http://www.w3.org/1999/xhtml">
<head>
<script src="https://ajax.googleapis.com/ajax/libs/jquery/3.3.1/jquery.min.js"></script>
<script
src="https://onlysharepoint2013.sharepoint.com/sites/Raju/SiteAssets/Preetihtml/LaptopJsFile.js"></script>
<link rel="stylesheet"
href="https://onlysharepoint2013.sharepoint.com/sites/Raju/SiteAssets/Preetihtml/LaptopStyle.css">
<meta charset="utf-8" />
<title>LAPTOP INFORMATION</title>
</head>
<body>
<p id="laptopinfo"></p>
</body>
</html>
```

JavaScript File

```
$(document).ready(function () {
ExecuteOrDelayUntilScriptLoaded(retrieveListItems, "sp.js");
});
var siteUrl = _spPageContextInfo.siteServerRelativeUrl;
var collListItem;
function retrieveListItems() {
var clientContext = new SP.ClientContext.get_current();
var oList = clientContext.get_web().get_lists().getByTitle('CompanyLaptopInfo');
var camlQuery = new SP.CamlQuery();
collListItem = oList.getItems(camlQuery);
```

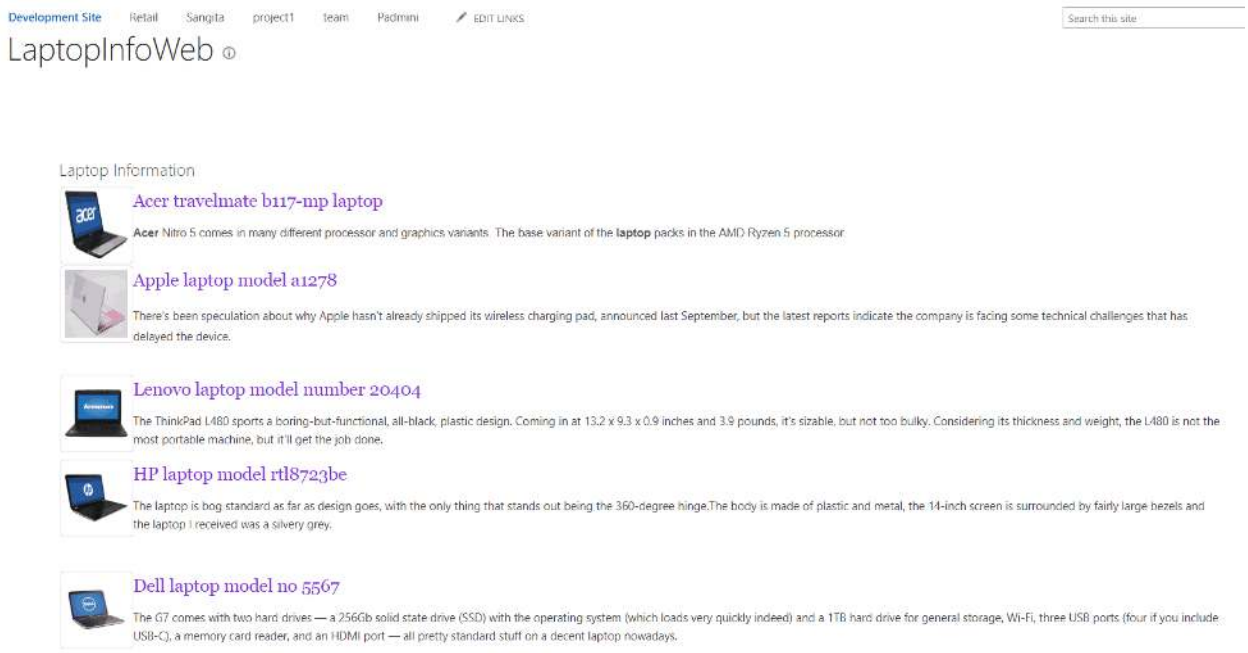
```
clientContext.load(collListItem);
clientContext.executeQueryAsync(Function.createDelegate(this,
this.onQuerySucceeded),
Function.createDelegate(this, this.onQueryFailed));
}
function onQuerySucceeded(sender, args) {
var listItemEnumerator = collListItem.getEnumerator();
var mytable = "<table>";
while (listItemEnumerator.moveNext()) {
var oListItem = listItemEnumerator.get_current();
var url =
"https://onlysharepoint2013.sharepoint.com/sites/Raju/Lists/CompanyLaptopInfo/DispFo
rm.aspx?ID=" + oListItem.get_item('ID');
var imgurl = oListItem.get_item('LaptopImage').get_url();
mytable += "<tr><td><img class=\'laptop\' src='" + imgurl + "'><a class=\'LaptopTitle\'
href='" + url + "'>" + oListItem.get_item('Title') + "</a><br /><p class=\'LaptopText\'>" +
oListItem.get_item('Description') + "</p></td></tr>";
}
mytable += "</table>";
$("#laptopinfo").html(mytable);
}
function onQueryFailed(sender, args) {
alert('Error: ' + args.get_message() + '\n' + args.get_stackTrace());
}
```

CSS File

```
img.laptop {
border: 1px solid #ddd;
```

```
border-radius: 4px;
padding: 5px;
width: 70px;
height: 75px;
float: left;
}
img.laptop:hover {
box-shadow: 0 0 2px 1px rgba(0, 140, 186, 0.5);
}
a.LaptopTitle {
color: blueviolet;
font-family: "Georgia", Times, serif;
font-size:20px;
}
a.LaptopTitle:visited {
color: yellowgreen;
}
a.LaptopTitle:hover {
color: red;
text-decoration: underline;
}
p.LaptopText {
font-size: 17px;
font-style: italic;
font-family: "Georgia", Times, serif;
}
```

Below is the output where we are seeing information in side a div as a tabular format from the list in SharePoint Online.

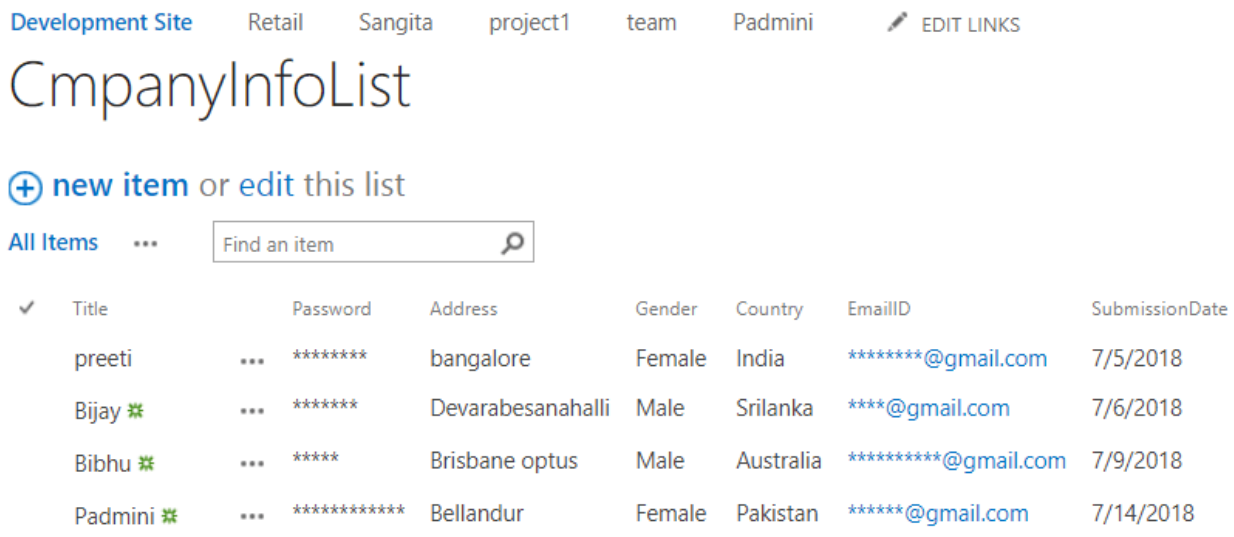


Example-46: Display SharePoint list data in HTML table using JavaScript

In this SharePoint JavaScript examples, how to display SharePoint list data in a HTML table using JavaScript object model in SharePoint Online or SharePoint 2013/2016/2019.

- User Name (single line text)
- Password(single line text)
- Address (Multiline textbox)
- Gender (Choice)
- Country (Choice / Dropdown list)
- Submission Date (Date and Time)

SharePointSky



Then add some Items into that List (CmpanyInfoList) like which I have shown in the below screenshot after that, we have to create an HTML File, JS File, and CSS File for table designing purpose.

Here I have created HTML File as (Mytable.html), JS File as (Mytable.js), and CSS File as (Mytablestyle.css) in the Visual Studio. The HTML code is given below:

```
<!DOCTYPE html>
<html lang="en" xmlns="http://www.w3.org/1999/xhtml">
<head>
<script src="https://ajax.googleapis.com/ajax/libs/jquery/3.3.1/jquery.min.js"></script>
<script
src="https://onlysharepoint2013.sharepoint.com/sites/Raju/SiteAssets/Preetihtml/Mytable.js"></script>
<link rel="stylesheet"
href="https://onlysharepoint2013.sharepoint.com/sites/Raju/SiteAssets/Preetihtml/Mytablestyle.css">
<meta charset="utf-8" />
```

```
<title></title>
</head>
<body>
<p id="ptitle"></p>
</body>
</html>
```

JS File

```
$(document).ready(function () {
ExecuteOrDelayUntilScriptLoaded(retrieveListItems, "sp.js");
});
var siteUrl = _spPageContextInfo.siteServerRelativeUrl;
var collListItem;
function retrieveListItems() {
var clientContext = new SP.ClientContext.get_current();
var oList = clientContext.get_web().get_lists().getByTitle('CmpanyInfoList');
var camlQuery = new SP.CamlQuery();
collListItem = oList.getItems(camlQuery);
clientContext.load(collListItem);
clientContext.executeQueryAsync(Function.createDelegate(this,
this.onQuerySucceeded),
Function.createDelegate(this, this.onQueryFailed));
}
function onQuerySucceeded(sender, args) {
var listItemEnumerator = collListItem.getEnumerator();
var mytable = "<table class=\"alternate\"><tr>" +
"<th>Title</th>" +
"<th>Address</th>" +
```

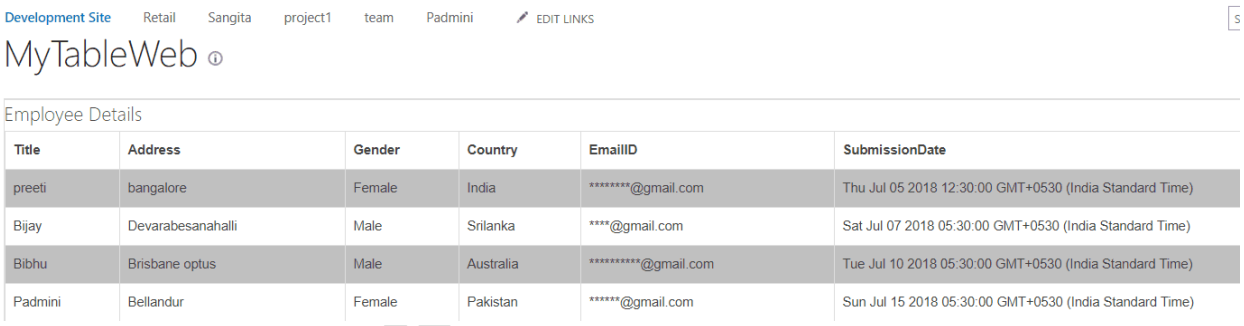
```
"<th>Gender</th>" +
"<th>Country</th>" +
"<th>EmailID</th>" +
"<th>SubmissionDate</th></tr>";
while (listItemEnumerator.moveNext()) {
var oListItem = listItemEnumerator.get_current();
mytable += "<tr><td>" + oListItem.get_item('Title') + "</td><td>" +
oListItem.get_item('Address') + "</td><td>" + oListItem.get_item('Gender') + "</td><td>"
+ oListItem.get_item('Country') + "</td><td>" + oListItem.get_item('EmailID') +
"</td><td>" + oListItem.get_item('SubmissionDate') + "</td></tr>";}
mytable += "</table>";
$("#ptitle").html(mytable);
}
function onQueryFailed(sender, args) {
alert('Error: ' + args.get_message() + '\n' + args.get_stackTrace());
}
```

CSS File

```
table {
font-family: arial, sans-serif;
border-collapse: collapse;
width: 100%;
}
td, th {
border: 1px solid #dddddd;
text-align: left;
padding: 8px;
}
```

```
.alternate tr:nth-child(2n) {
background-color: silver;
}
.alternate tr {
background-color: white;
}
.alternate tr:nth-child(2n):hover, .alternate tr:hover {
background-color: grey;
}
```

Below is the SharePoint list and the final HTML tabular formatted data.



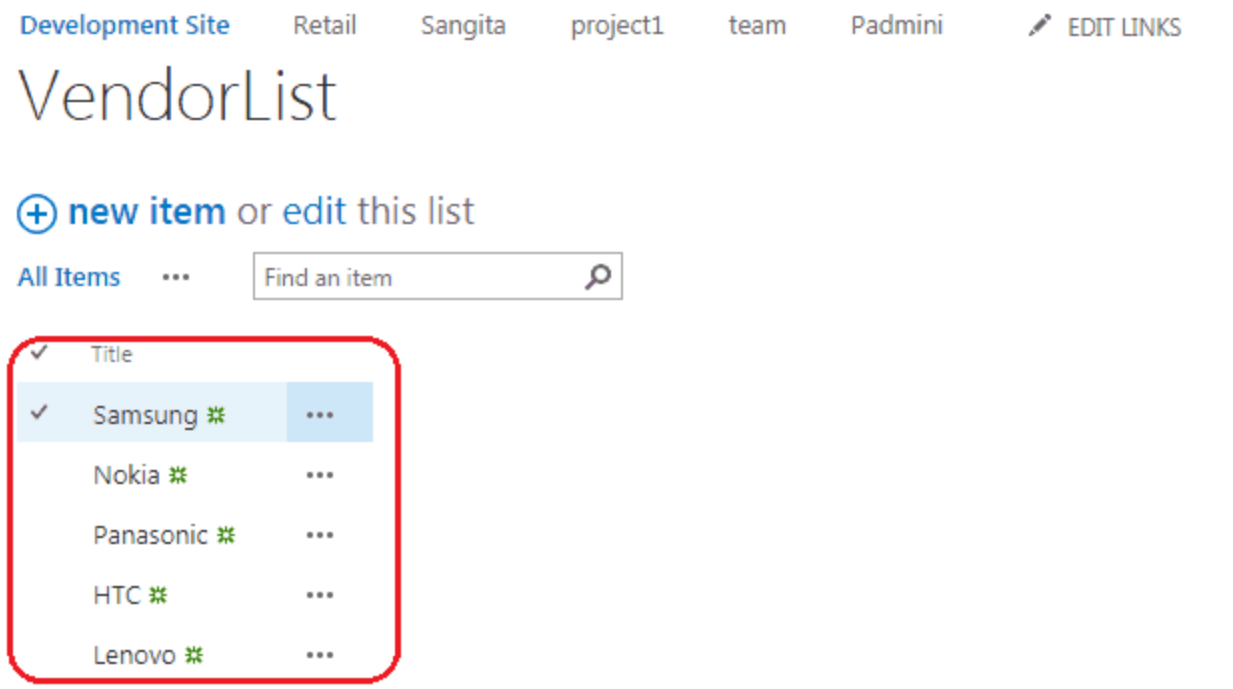
Title	Address	Gender	Country	EmailID	SubmissionDate
preeti	bangalore	Female	India	*******@gmail.com	Thu Jul 05 2018 12:30:00 GMT+0530 (India Standard Time)
Bijay	Devarabesanahalli	Male	Srilanka	****@gmail.com	Sat Jul 07 2018 05:30:00 GMT+0530 (India Standard Time)
Bibhu	Brisbane optus	Male	Australia	*********@gmail.com	Tue Jul 10 2018 05:30:00 GMT+0530 (India Standard Time)
Padmini	Bellandur	Female	Pakistan	******@gmail.com	Sun Jul 15 2018 05:30:00 GMT+0530 (India Standard Time)

Example-47: Bind SharePoint list items to dropdownlist using javascript object model (jsom) in SharePoint Online

In this SharePoint SharePoint JavaScript example, I will show how to bind SharePoint list items to dropdown list using JavaScript.

I have a SharePoint Online list, name as “VendorList” which has a Title column and it contains below items:

- Samsung
- Nokia
- Panasonic
- HTC
- Lenovo



HTML Code

```
<!DOCTYPE html>
<html lang="en" xmlns="http://www.w3.org/1999/xhtml">
<head>
<script src="https://ajax.googleapis.com/ajax/libs/jquery/3.3.1/jquery.min.js"></script>
<script
src="https://onlysharepoint2013.sharepoint.com/sites/Raju/SiteAssets/Preetihtml/VendorJsFile.js"></script>
<link rel="stylesheet"
href="https://onlysharepoint2013.sharepoint.com/sites/Raju/SiteAssets/Preetihtml/VendorStyle.css">
<meta charset="utf-8" />
<title>Vendor List</title>
</head>
<body>
Vendor List: <select name="Vendor" id="MobileVendor"></select>
```

```
</body>
</html>
```

JS File

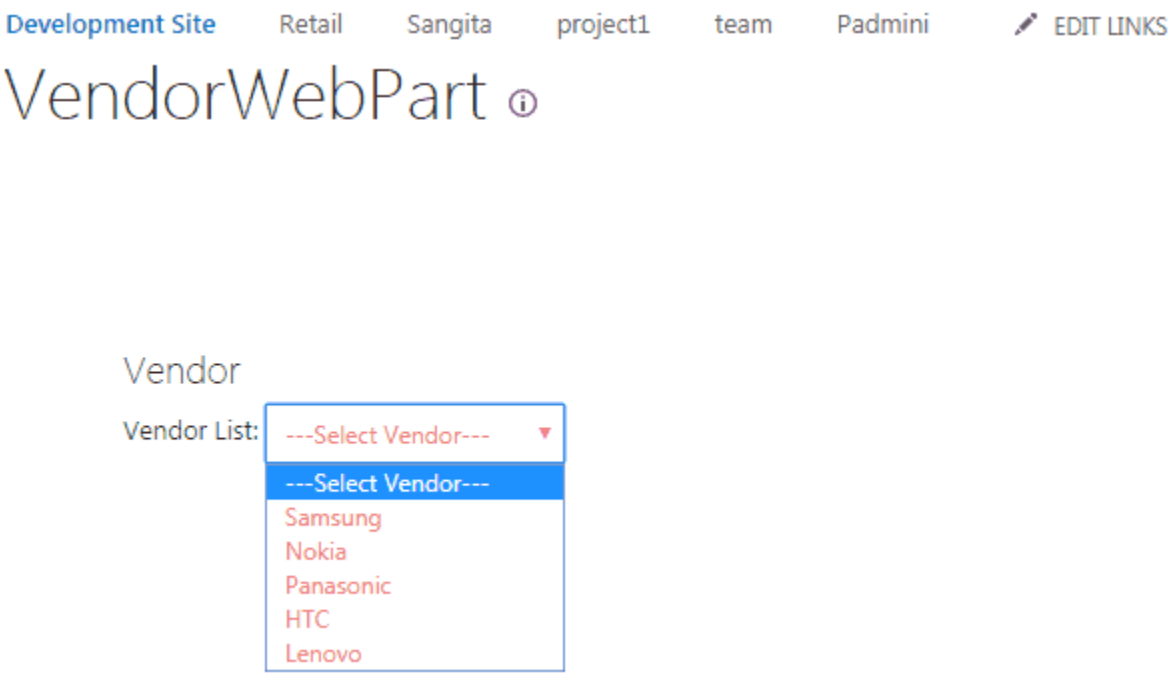
```
$(document).ready(function () {
ExecuteOrDelayUntilScriptLoaded(Bindvendor, "sp.js");
});
var collListItem;
function Bindvendor() {
var clientContext = new SP.ClientContext.get_current();
var oList = clientContext.get_web().get_lists().getByTitle('VendorList');
var camlQuery = new SP.CamlQuery();
collListItem = oList.getItems(camlQuery);
clientContext.load(collListItem);
clientContext.executeQueryAsync(Function.createDelegate(this,
this.onQuerySucceeded),
Function.createDelegate(this, this.onQueryFailed));
}
function onQuerySucceeded(sender, args) {
var listItemEnumerator = collListItem.getEnumerator();
$("#MobileVendor").append('<option>---Select Vendor---</option>')
while (listItemEnumerator.moveNext()) {
var oListItem = listItemEnumerator.get_current();
$("#MobileVendor").append('<option>' + oListItem.get_item('Title') + '</option>')
}
}
function onQueryFailed(sender, args) {
alert('Error: ' + args.get_message() + '\n' + args.get_stackTrace());
```

```
}
}
```

CSS File

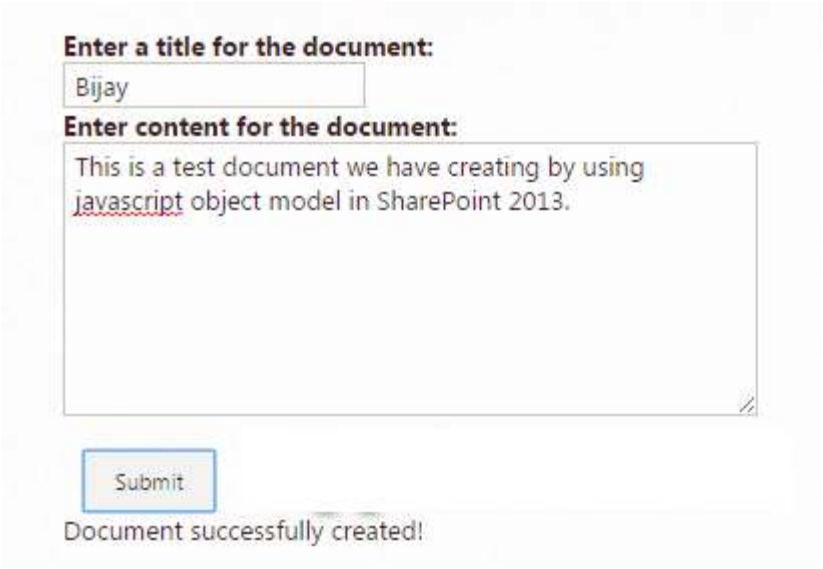
```
#MobileVendor {
padding: 5px;
color: lightcoral;
font-size: 12px;
width: 150px;
}
```

Below is the final dropdown control with all the vales binded.



Example-48: Create file or document using JavaScript object model (jsom) in SharePoint Online Office 365

Here we will see how to create a file or document that will be stored in a document library using JSOM (javascript object model) in SharePoint Online.



HTML Code

```
<div id="CreateFile">
<div>
<strong>Enter a title for the document:</strong>
<br />
<input type="text" id="txtDocumentTitle" />
</div>
<div>
<strong>Enter content for the document:</strong>
<br />
<textarea cols="20" id="txtDocumentContent"></textarea>
</div>
<br />
```

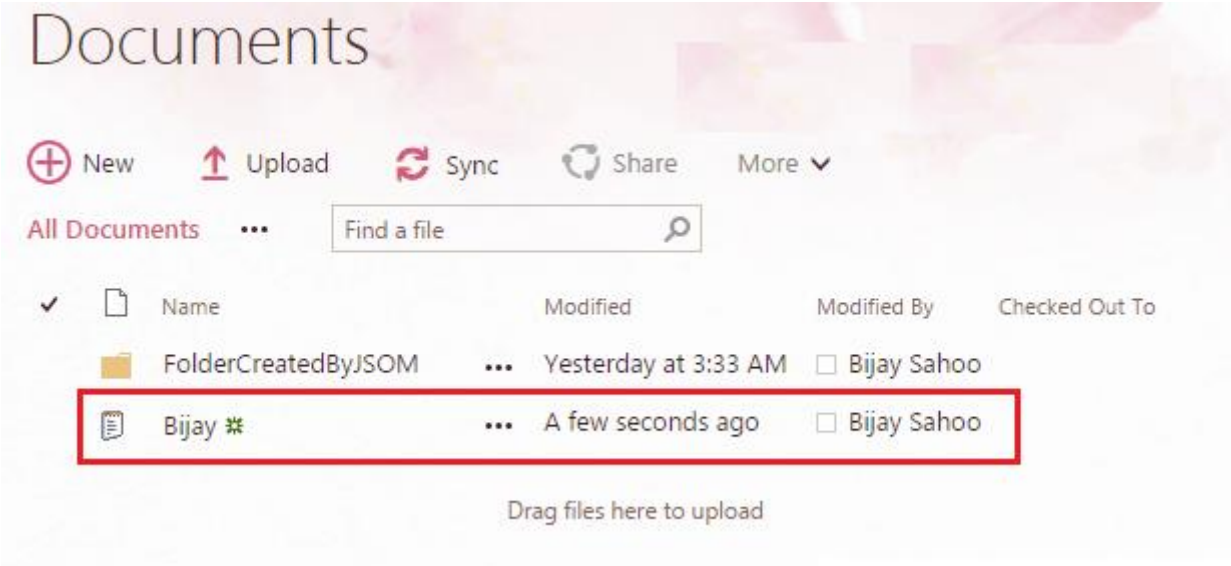
```
<input type="button" id="btnSubmit" value="Submit" />
</div>
<div id="divResults"></div>
```

JSOM Code

```
<script src="https://ajax.googleapis.com/ajax/libs/jquery/1.11.2/jquery.min.js"></script>
<script>
$(function () {
bindButtonClick();
});
function bindButtonClick() {
$("#btnSubmit").on("click", function () {
createDocument();
});
}
function createDocument() {
var docTitle = $("#txtDocumentTitle").val() + ".txt";
var docContent = $("#txtDocumentContent").val();
var clientContext = new SP.ClientContext();
var oWebsite = clientContext.get_web();
var oList = oWebsite.get_lists().getByTitle("Documents");
var fileCreateInfo = new SP.FileCreationInformation();
fileCreateInfo.set_url(docTitle);
fileCreateInfo.set_content(new SP.Base64EncodedByteArray());
for (var i = 0; i < docContent.length; i++) {
fileCreateInfo.get_content().append(docContent.charCodeAt(i));
}
this.newFile = oList.get_rootFolder().get_files().add(fileCreateInfo);
```

```
clientContext.load(this.newFile);
clientContext.executeQueryAsync(
Function.createDelegate(this, this.onQuerySucceeded),
Function.createDelegate(this, this.onQueryFailed)
);
}
function onQuerySucceeded() {
$("#divResults").html("Document successfully created!");
}
function onQueryFailed(sender, args) {
alert('Request failed. ' + args.get_message() +
'\n' + args.get_stackTrace());
}
</script>
```

Now if you will open SharePoint document library, you can see the document created in the Document library in SharePoint Online.



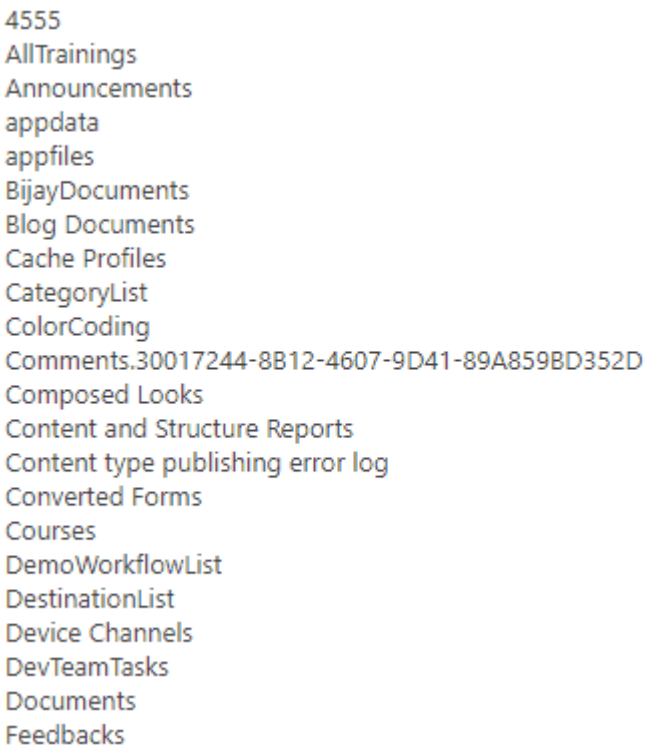
Example-49: Retrieve all lists and libraries from SharePoint site using JavaScript object model (JSOM)

In this SharePoint JavaScript examples, we will fetch all the list and document library information using jsom (JavaScript object model) in SharePoint online.

```
<script src="https://ajax.googleapis.com/ajax/libs/jquery/1.11.2/jquery.min.js"></script>
<div id="divGetListData"></div>
<script>
$(function () {
ExecuteOrDelayUntilScriptLoaded(getAllLists, "sp.js");
});
function getAllLists() {
var clientContext = new SP.ClientContext();
var oWebsite = clientContext.get_web();
this.collList = oWebsite.get_lists();
clientContext.load(collList);
clientContext.executeQueryAsync(
Function.createDelegate(this, this.onQuerySucceeded),Function.createDelegate(this,
this.onQueryFailed)
);
}
function onQuerySucceeded() {
var listInfo = "";
var listEnumerator = collList.getEnumerator();
while (listEnumerator.moveNext()) {
var oList = listEnumerator.get_current();
listInfo += oList.get_title() + '<br />';
}
```

```
$("#divGetListData").html(listInfo);
}
function onQueryFailed(sender, args) {
alert('Request failed. ' + args.get_message() +
'\n' + args.get_stackTrace());
}
</script>
```

Below it is showing complete count and the List and Documents in SharePoint.



Example-50: Retrieve all SharePoint groups using JSOM (JavaScript object model)

This jsom SharePoint example we will discuss how to retrieve all SharePoint groups in SharePoint Online using JSOM (JavaScript object model).

```
<input type='button' value='Get All Groups' onclick="clickMethod();"/>
<br />
<script language="javascript" type="text/javascript">
var siteGroups ='';
function clickMethod() {
var clientContext = new SP.ClientContext.get_current();
siteGroups = clientContext.get_web().get_siteGroups();
clientContext.load(siteGroups);
clientContext.executeQueryAsync(onQuerySucceeded, onQueryFailed);
}
function onQuerySucceeded() {
var allGroups='Group Name:       Group ID '+'\n';
for (var i =0 ; i < siteGroups.get_count(); i++)
{
allGroups +=siteGroups.itemAt(i).get_title()+'       '+siteGroups.itemAt(i).get_id()+'\n';
}
alert(allGroups);
}
function onQueryFailed() {
alert('Request failed.');
}
</script>
```

Once you run the code, you should be able to see all SharePoint groups in SharePoint Online Office 365.

Example-51: Deferred and Promise in JavaScript Object Model in SharePoint 2013 to make asynchronous to synchronous call

This SharePoint JavaScript examples, I will show how to use Deferred and Promise in JSOM (JavaScript Object Model) in SharePoint 2013 to make asynchronous to synchronous call.

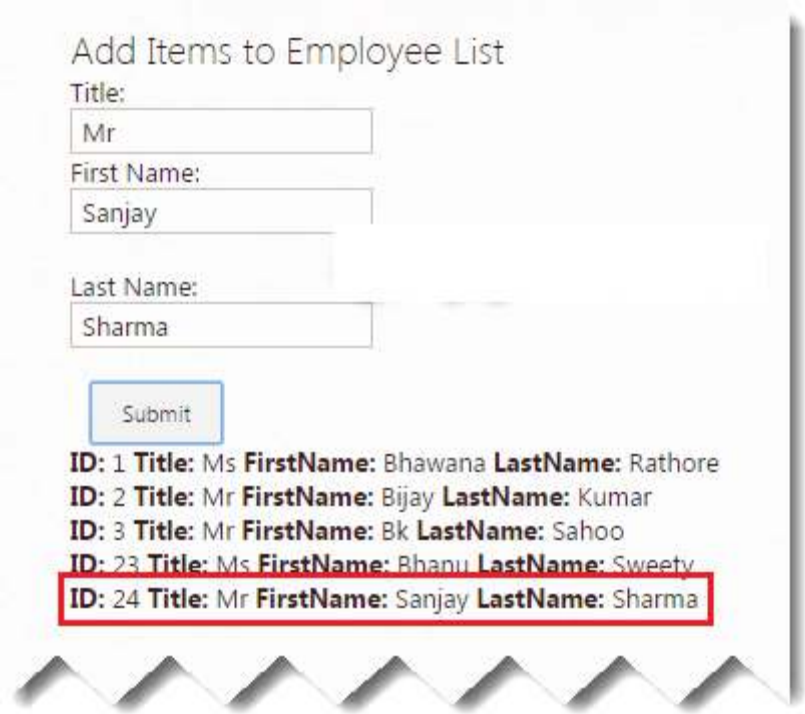
```
<p>Add Items to Employee List</p>
<div id="AddListData">
<div>
Title:
<br />
<input type="text" id="txtTitle" />
</div>
<div>
First Name:
<br />
<input type="text" id="txtFirstName" />
</div>
<br />
<div>
Last Name:
<br />
<input type="text" id="txtLastName" />
</div>
<br />
<div>
<input id="btnSubmit" type="button" value="Submit" />
</div>
```

```
</div>
<div id="divResult"></div>
<div id="divListItems"></div>
<script src="https://ajax.googleapis.com/ajax/libs/jquery/1.11.2/jquery.min.js"></script>
<script>
var d = $.Deferred();
var listItemInfo = '';
$(function () {
bindButtonClick();
});
function bindButtonClick() {
$("#btnSubmit").on("click", function () {
var p = addListItem();
p.done(function (result) {
retrieveListItems();
});
p.fail(function (result) {
alert('error');
});
});
}
function addListItem() {
var title = $("#txtTitle").val();
var firstname = $("#txtFirstName").val();
var lastname = $("#txtLastName").val();
var clientContext = new SP.ClientContext();
var oList = clientContext.get_web().get_lists().getByTitle('Employees');
var itemCreateInfo = new SP.ListItemCreationInformation();
```

```
this.oListItem = oList.addItem(itemCreateInfo);
oListItem.set_item('Title', title);
oListItem.set_item('FirstName', firstname);
oListItem.set_item('LastName', lastname);
oListItem.update();
clientContext.load(oListItem);
clientContext.executeQueryAsync(
Function.createDelegate(this, this.onAddSucceeded),Function.createDelegate(this,
this.onAddFailed)
);
return d.promise();
}
function onAddSucceeded(sender, args) {
d.resolve(sender, args);
}
function onAddFailed(sender, args) {
d.reject(sender, args);
}
function retrieveListItems() {
var clientContext = new SP.ClientContext();
var oList = clientContext.get_web().get_lists().getByTitle('Employees');
var camlQuery = new SP.CamlQuery();
this.collListItem = oList.getItems(camlQuery);
clientContext.load(collListItem);
clientContext.executeQueryAsync(Function.createDelegate(this,
this.onQuerySucceeded),Function.createDelegate(this, this.onQueryFailed)
);
}
```

```
function onQuerySucceeded(sender, args) {
listItemInfo = '';
var listItemEnumerator = collListItem.getEnumerator();
while (listItemEnumerator.moveNext()) {
var oListItem = listItemEnumerator.get_current();
listItemInfo += '<strong>ID: </strong> ' + oListItem.get_id() +
' <strong>Title:</strong> ' + oListItem.get_item('Title') +
' <strong>FirstName:</strong> ' + oListItem.get_item('FirstName') +
' <strong>LastName:</strong> ' + oListItem.get_item('LastName') +
'<br />';
}
$("#divListItems").html(listItemInfo);
}
function onQueryFailed(sender, args) {
alert('Hello');
}
</script>
```

Share



Conclusion

In this tutorial, we learned top 51 SharePoint JavaScript examples, and also you can download a PDF which contains all the SharePoint JSOM (JavaScript Object Model) examples.

This JSOM SharePoint example works with SharePoint Online as well as SharePoint 2019, SharePoint 2016, and SharePoint 2013 also.