@pnp/sp/webs

Webs are one of the fundamental entry points when working with SharePoint. Webs serve as a container for lists, features, sub-webs, and all of the entity types.

Add a Web

Using the library you can add a web to another web's collection of subwebs. The basic usage requires only a title and url. This will result in a team site with all of the default settings.

import { sp, WebAddResult } from "@pnp/sp";

sp.web.webs.add("title", "subweb1").then((w: WebAddResult) => {

// show the response from the server when adding the web

console.log(w.data);

w.web.select("Title").get().then(w => {

// show our title

console.log(w.Title);

});

});

You can also provide other settings such as description, template, language, and inherit permissions.

import { sp, WebAddResult } from "@pnp/sp";

// create a German language wiki site with title, url, description, which inherits permissions

sp.web.webs.add("wiki", "subweb2", "a wiki web", "WIKI#0", 1031, true).then((w: WebAddResult) => {

// show the response from the server when adding the web

console.log(w.data);

w.web.select("Title").get().then(w => {

// show our title

console.log(w.Title);

});

});

Create Default Associated Groups

If you create a web that doesn't inherit permissions from the parent web, you can create its default associated groups (Members, Owners, Visitors) with the default role assigments (Contribute, Full Control, Read)

import { sp, WebAddResult } from "@pnp/sp";

sp.web.webs.add("title", "subweb1", "a wiki web", "WIKI#0", 1031, false).then((w: WebAddResult) => {

w.web.createDefaultAssociatedGroups().then(() => {

// ...

});

});

Get A Web's properties

import { sp } from "@pnp/sp";

// basic get of the webs properties

sp.web.get().then(w => {

console.log(w.Title);

});

// use odata operators to get specific fields

sp.web.select("Title").get().then(w => {

console.log(w.Title);

});

// use with get to give the result a type

sp.web.select("Title").get<{ Title: string }>().then(w => {

console.log(w.Title);

});

Get Complex Properties

Some properties, such as AllProperties, are not returned by default. You can still access them using the expand operator.

import { sp } from "@pnp/sp";

sp.web.select("AllProperties").expand("AllProperties").get().then(w => {

console.log(w.AllProperties);

});

Get a Web Directly

You can also use the Web object directly to get any web, though of course the current user must have the necessary permissions. This is done by importing the web object.

import { Web } from "@pnp/sp";

let web = new Web("https://my-tenant.sharepoint.com/sites/mysite");

web.get().then(w => {

console.log(w);

});

Open Web By Id

Because this method is a POST request you can chain off it directly. You will get back the full web properties in the data property of the return object. You can also chain directly off the returned Web instance on the web property.

sp.site.openWebById("111ca453-90f5-482e-a381-cee1ff383c9e").then(w => {

//we got all the data from the web as well

console.log(w.data);

// we can chain

w.web.select("Title").get().then(w2 => {

// ...

});

});

Update Web Properties

You can update web properties using the update method. The properties available for update are listed in [this table](https://msdn.microsoft.com/en-us/library/office/dn499819.aspx#bk_WebProperties). Updating is a simple as passing a plain object with the properties you want to update.

import { Web } from "@pnp/sp";

let web = new Web("https://my-tenant.sharepoint.com/sites/mysite");

web.update({

Title: "New Title",

CustomMasterUrl: "{path to masterpage}",

Description: "My new description",

}).then(w => {

console.log(w);

});

Delete a Web

import { Web } from "@pnp/sp";

let web = new Web("https://my-tenant.sharepoint.com/sites/mysite");

web.delete().then(w => {

console.log(w);

});