SharePoint Framework for Administrators

Erwin van Hunen
Waldek Mastykarz

Rencore

SharePoint Framework differs from other development models available to extend SharePoint. This document should serve as a comprehensive list of things to consider before deploying the first SharePoint Framework solution in your production tenant.

URL: www.rencore.com

Email: sales@rencore.com

Twitter: @rencore - @erwinvanhunen - @waldekm

LinkedIn: https://www.linkedin.com/in/rencore/

Contents

1. What is SharePoint Framework

SharePoint Framework is a new model for extending SharePoint. Using the SharePoint Framework, developers can build web parts as well as extensions. SharePoint Framework extensions can add functionality to the header or the footer of the page, can expose additional menu options in lists and document libraries or control how the data in lists and document libraries is displayed.

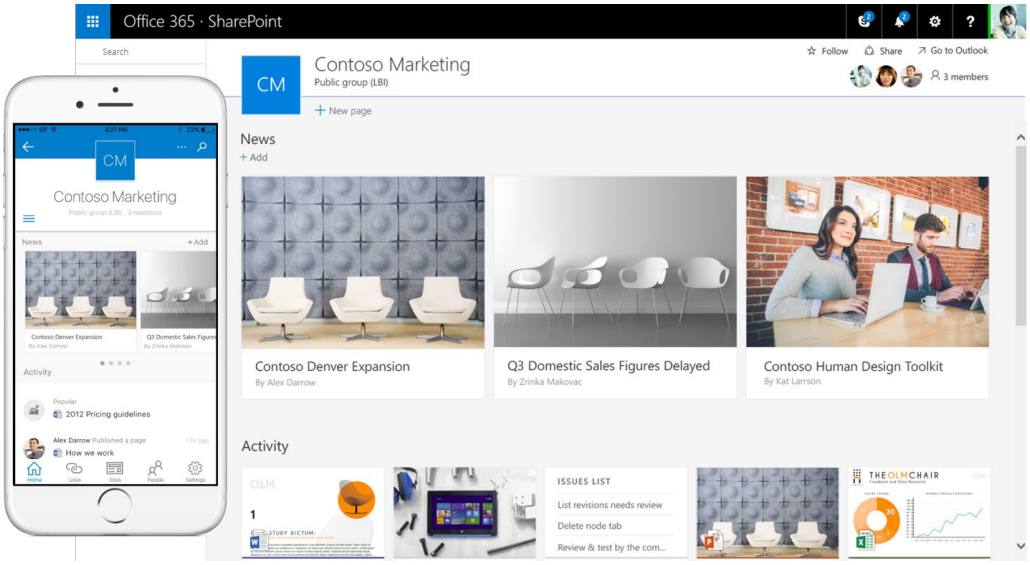


Figure 1.

Modern SharePoint user experience extended with SharePoint Framework solutions

SharePoint Framework solutions are built with JavaScript. When a SharePoint Framework solution is deployed to a site, SharePoint loads JavaScript that is a part of this solution on the page and executes it each time a user visits the page.

1.1. Why another SharePoint development model

Over the years, SharePoint customization models changed to guarantee platform hygiene and stability. In the past, SharePoint solutions were deployed to SharePoint Farms. Therefore, a poorly written solution could bring down the entire farm. With SharePoint Add-ins and SharePoint Framework solutions, the code is either running in the web browser or on an external server and has significantly less impact on the Farm than Farm solutions.

Extending SharePoint using JavaScript isn't new. For years developers and power users could use the Content Editor- and Script Editor Web Parts to embed pieces of JavaScript on pages to add new functionality. The downside of this approach was, that the IT had no means to track the usage of these customizations across the organization and couldn't assist users or maintain these solutions. SharePoint Framework solutions are deployed through the app catalog. This allows organizations to keep track of which solutions are being deployed to their SharePoint environment and what building blocks they consist of.

1.2. How does SharePoint Framework compare to other development models

SharePoint Framework doesn't replace any of the existing SharePoint development models. In fact, in many cases, SharePoint Framework solutions will be accompanied by solutions built using other SharePoint development models to allow server-side processing for handling events or building long-running operations or scheduled tasks which cannot be done reliably in a web browser.

SharePoint Framework solutions consist of JavaScript scripts that are executed in the web browser as a part of SharePoint pages. As such, scripts that load as a part of a SharePoint Framework solution inherit the security context of the currently signed-in user. In other words, whatever the currently signed-in user is allowed to do on the intranet, any of the scripts loaded on the page can do as well. At this moment, there is no way to allow a SharePoint Framework solution to perform only specific operations.

Tenant administrators can allow SharePoint Framework solutions to connect to specific APIs secured with Azure Active Directory. When connecting to these APIs, SharePoint Framework will automatically facilitate the authentication on behalf of the current user.

2. Considerations for your organization before you start using SharePoint Framework

SharePoint Framework differs from other development models available to extend SharePoint. If your organization is new to the SharePoint Framework, there are a number of things you should consider before deploying the first SharePoint Framework solution in your production tenant.

2.1. Client-side instead of managed code

While solutions built using other SharePoint development models could contain some JavaScript, the majority of their code was written in .NET and running on a server. SharePoint Framework solutions, on the contrary, consist exclusively of JavaScript that runs in the web browser. While SharePoint Framework solutions can communicate with other APIs that run on another server and are built using the SharePoint Add-in model, these APIs are deployed and managed separately and are not part of a SharePoint Framework solution package.

Because the solution's code runs in the web browser of the user who interacts with it, there is no central place where the code can be monitored by default. When building solutions, developers must include an ability to track how their code works and to log any issues that might occur including all the data about the environment in which the code was running (operating system, web browser, etc.) in order to be able to debug these issues.

2.2. Different JavaScript frameworks and libraries

SharePoint Framework is based on open-source tooling that resembles what developers outside of the Microsoft ecosystem use. Microsoft provides developers only with the tools to create and build SharePoint Framework solutions. Developers are free to choose which building blocks they want to use in their solutions. There are over 650.000 packages available on the internet that developers could use in their SharePoint Framework solutions. The quality of these packages, how widely they're adopted in the market as well as the license under which they're distributed vary per package. To simplify the maintenance of your solutions and avoid costly changes, your organization should decide which libraries are allowed to be used. Before deploying a solution to production, you should ensure that no other libraries are being used in the solution.

2.3. Assets can be hosted anywhere

When building SharePoint Framework solutions, developers decide where the scripts that build up the solution are hosted. Developers can choose to include the assets in the solution package and have them deployed to the same Office 365 tenant where the solution will be used. But they can also choose to load the scripts from any arbitrary location on the internet. If the scripts are hosted on a location that isn't managed by your organization, it exposes your organization to a number of risks. Before deploying a SharePoint Framework solution to your tenant, you should ensure that its scripts are hosted in a location that your organization manages.

2.4. Communicating with APIs secured with Azure AD

Some SharePoint Framework solutions connect to APIs secured with Azure AD, such as the Microsoft Graph, or enterprise APIs owned by your organization, to retrieve their data. Tenant administrators can decide which APIs SharePoint Framework solutions should be allowed to connect to. But here is the catch. When you choose an API to be available to SharePoint Framework solutions, it can also be used by any piece of JavaScript running in your tenant. Every JavaScript widget, jQuery carousel and weather widget embedded using Content Editor Web Part in SharePoint, could potentially connect to these APIs on behalf of the currently signed-in user. So, before you choose APIs that should be available to client-side solutions in your tenant, you should carefully evaluate what data will be available and what scripts you have running in your tenant.

2.5. Organization readiness checklist

SharePoint Framework solutions use the existing SharePoint infrastructure and where necessary extend it with additional capabilities. To use them effectively in your organization, the following is a list of things that you and your organization should take into account.

2.5.1. Setup SharePoint app catalog

SharePoint Framework solutions are deployed through the app catalog. To be able to use them, you have to setup an app catalog site in your tenant.

2.5.2. Decide who is allowed to deploy SharePoint Framework solutions

All SharePoint Framework solutions must be deployed through the app catalog. You should decide who in your organization should be allowed to deploy solutions in your tenant. Allowing solutions to be deployed only by tenant admins gives you full control of what solutions are being used in the tenant. Allowing other users to deploy solutions might help you scale the solution approval process but it's also not without risk. Remember, all SharePoint Framework solutions run as a part of the page and can access all information in the current tenant which the currently signed-in user can access.

2.5.3. Decide if you want to use site collection app catalogs

In SharePoint Online, tenant administrators can enable app catalogs on specific site collections. This allows you to delegate the approval and deployment process of solutions to site collection administrators. Solutions deployed in the site collection app catalog can be used only in that particular site collection but when executed on the page, they can still access information from the whole tenant and APIs on behalf of the current user.

You should decide if you want to use site collection app catalogs in your tenant and if so, under which conditions it will be enabled. You might for example allow it only for sites of specific type, classification or not at all.

2.5.4. Choose a hosting location

SharePoint Framework solutions consist of a manifest and JavaScript files that contain the actual code. These JavaScript files must be located on a web server and must be accessible by each user signed into SharePoint or the solution will fail.

You have a number of options to choose from with regards to where your SharePoint Framework solutions scripts should be hosted that range from using the free Office 365 Public CDN to a custom server managed by your organization. Each location has its advantages and disadvantages. The important part for you to remember, is that developers choose where the files will be deployed when building the solution. Before actually deploying the solution, you should verify the location they chose is the location your organization decided to use.

2.5.5. Decide how you want to handle API permission requests

SharePoint Framework solutions can communicate with APIs secured with Azure AD. These can be APIs provided by Microsoft, such as the Microsoft Graph, or enterprise APIs built by your organization. Before any of these APIs can be used by a SharePoint Framework solution, it needs to be approved for use by the tenant administrator.

When building SharePoint Framework solutions, developers can include API permission requests in their packages. When you deploy a package with a permission request to the app catalog, SharePoint will prompt you to review the requested permissions and either approve the request or deny it. Even though an API permission request is issued by a solution, it applies to the whole tenant. So rather than having solutions issue request API permissions, you could decide which APIs should be available in your tenant and approve their usage upfront. This would prevent you from having to reactively respond to each permission request, but to do effectively, you would have to think upfront which APIs should be available in your tenant and what risks their usage introduces to your confidential information.

3. Anatomy of SharePoint Framework solutions

SharePoint Framework solutions are client-side solutions that consist exclusively of JavaScript and are distributed as .sppkg files. Because solution packages are in fact ZIP archives, you can easily extract and examine their contents to verify them before deploying them to your production tenant.

3.1. What is inside a solution package

Each solution package contains one or more SharePoint Feature, one or more manifests of SharePoint Framework web parts and extensions and, optionally, JavaScript files with the solution's code. When building solutions, developers can decide if they want to include the script files in the solution package or if they should be deployed elsewhere. If the script files are included in the solution package, they will be deployed to the SharePoint app catalog. There is no restriction where the code files are deployed to, as long as they are accessible by the end-users via a URL.

3.2. Resemblance to other solution package formats

SharePoint Framework solution packages resemble SharePoint add-ins. Similarly, they are deployed to the app catalog and can be installed in a specific site or to all sites at once, even though the exact process is slightly different.

3.3. How it works

After deploying a SharePoint Framework solution package, the web parts and extensions it contains can be used on sites. Extensions are activated either via Features included in the solution package or programmatically through custom actions. End-users can add web parts to pages themselves. When opening a page with a SharePoint Framework component on it, SharePoint will retrieve the corresponding manifest and from it the URL where the code file is located. Next, it will download that file and execute it.

Unlike SharePoint add-ins, SharePoint Framework solutions are being executed as a part of the page. There is no separate permission layer around SharePoint Framework solutions and every piece of script can perform all tasks that the current user is allowed to. This is why you should carefully assess each solution and its contents, before you deploy them to your tenant.

4. Deploying and managing SharePoint Framework solutions

Until recently, it was not possible to programmatically upload and install your solutions for SharePoint Online. With the availability of the Application Lifecycle APIs (ALM) it is now possible to fully automate this process.

We have to understand the various states of solution: deployed and installed. The difference between the two is that a deployed app is made available to your users for installation, where as an installed solution is actually added to a site to be used. The process of deploying a solution means that you make it available for people to add the solution to a site.

4.1. Selecting your scope

It is important that you first select the scope to which you want to make your solution available. SharePoint Online provides two scopes: site collection and tenant. If you want to make your solution available for one site collection only you will have to activate the site collection scoped app catalog for that site. Check out https://docs.microsoft.com/en-us/sharepoint/dev/general-development/site-collection-app-catalog for more information.

4.2. Deploying solutions

Before you can start using the particular SharePoint Framework solution in your organization, you have to deploy it to the app catalog in your tenant.

4.2.1. Manually deploying solution

After selecting your scope, navigate to the appropriate app catalog. In the examples below we decided to upload the solution to the tenant app catalog.

To find the location of the app catalog, navigate to your Tenant Administration site, usually located at http://<yourtenant>-admin.sharepoint.com (Figure 2.).

Navigate to ‘apps’ in the left navigation and select ‘App Catalog’ in the resulting page. You will either be given the option to create an app catalog, or if already present, you will be redirected to your app catalog site for your tenant.

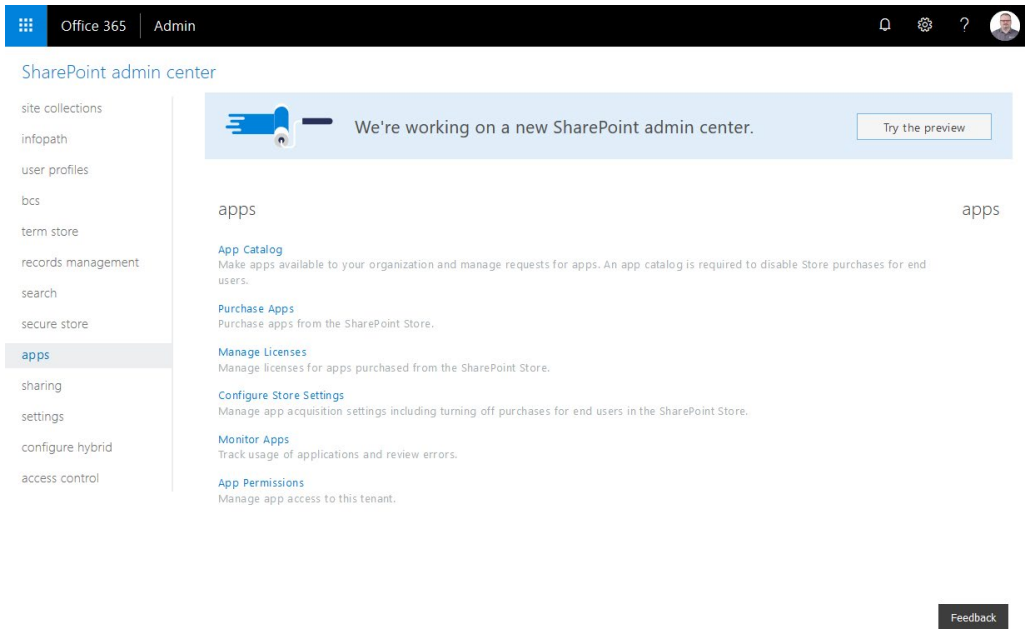


Figure 2. Tenant Administration site (http://<yourtenant>-admin.sharepoint.com)

Select 'Apps for SharePoint' in the left-hand navigation (Figure 3.).

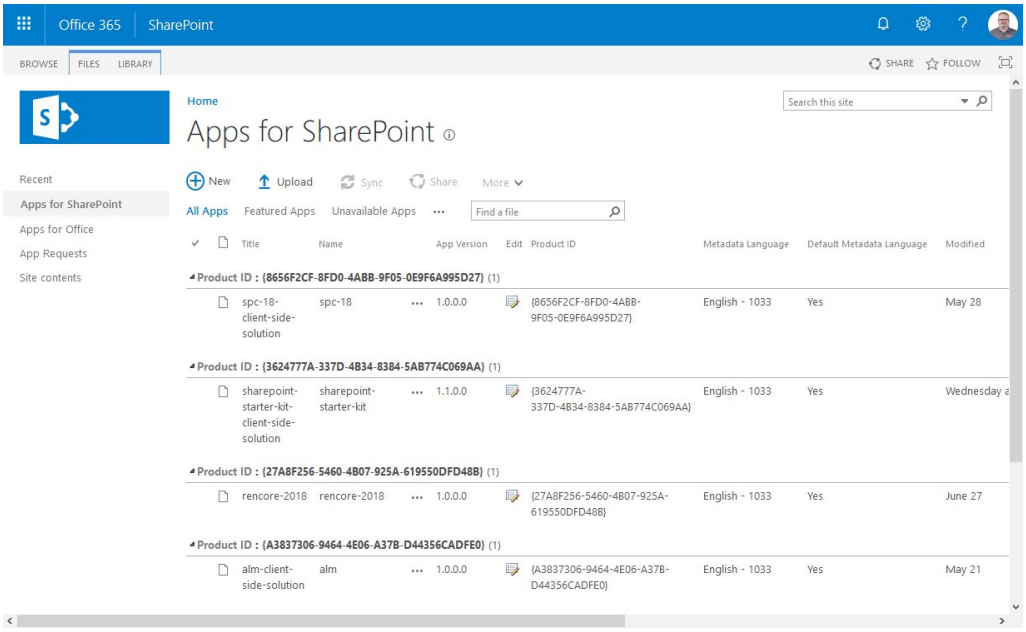


Figure 3. Apps for SharePoint

In certain scenarios you will be asked to provide a consent for the application to use certain APIs -answer those appropriately. Make sure to consult with the developer of the solution to understand why the solution requires access to these APIs. In certain scenarios an application could require access to all sites through the Microsoft Graph for instance. If you grant the application access to this, keep in mind that you are granting ALL other solutions in your solution access to all sites.

4.2.2. Deploying a solution with PnP PowerShell

Getting Started

PnP PowerShell is an open source and community-maintained PowerShell module that allows you to perform maintenance to your sites and solutions. These extend the functionality of the Microsoft provided SharePoint Online Management Shell. If you have not installed the cmdlet module yet, the easiest way to install it is by executing the following cmdlet:

```
Install-Module -Name SharePointPnPPowerShellOnline -Scope CurrentUser
```

This will download the module from the Microsoft hosted PowerShell Gallery (https://www.powershellgallery.com) and install it in your environment. Alternatively you can download an offline installer from https://github.com/SharePoint/PnP-PowerShell/releases/latest.

Make sure to restart PowerShell after installing the module.

The next step is to connect to your tenant with the **`Connect-PnPOnline cmdle`**t:

```
Connect-PnPOnline -Url https://<yourtenant>.sharepoint.com
```

You will be prompted to provide a username and password. Make sure to authenticate as a tenant administrator.

After connecting you can easily upload your solution by using the **`Add-PnP cmdlet`**.

Deploying a solution

To deploy a solution, simply type

```
Add-PnPApp -Path <pathtoyoursolution.sppkg>
```

Notice that the **`Add-PnPApp`** *has more parameters, for instance* **`-Scope`** *which allows you to upload a package to a site collection app catalog. In order for this to work, make sure to connect to the actual site where you want to upload the solution with* **`Connect-PnPOnline`**. *In PowerShell enter '*`Get-Help Add-PnPApp -Detailed`*' to see a description of all parameters and a few examples.*

4.2.3. Deploying a solution with the Office 365 CLI

Getting Started

The Office 365 CLI is an open source and community-maintained command line interface which runs on top of Node.js runtime. In order to be able to run this, you are required to have Node.js installed. See https://nodejs.org/ for more information about Node.js and how to install it.

After you have successfully installed Node.js, it is straightforward to install the Office 365 CLI:

```
npm i -g @pnp/office365-cli
```

Open a command line prompt (which can be PowerShell of Cmd.exe) and enter

```
office365
```

This will launch an interactive session allowing you to upload the solution. First, like with PnP PowerShell, you will have to connect to your tenant:

```
spo connect https://<yourtenant>-admin.sharepoint.com
```

Follow the steps provided to authenticate against your tenant and make sure you authenticate as a tenant administrator.

Deploying a solution

To deploy a solution, simply type

```
spo app add --filePath <pathtoyoursolution.sppkg>
```

Notice that the CLI command has more parameters, for instance `--overwrite` *which allows you to overwrite an already existing solution in your app catalog. In the CLI enter '*`help spo app add`*' to a detailed overview of all parameters and a few examples.*

4.3. Installing solutions

Once the solution has been deployed in the app catalog, it needs to be installed in sites. If the particular solution doesn't support global deployment or you chose to only enable it in specific sites, you need to install it in every site where it should be available to users.

4.3.1. Manually installing solutions

You can install a solution manually by navigating to the site, and select 'Add an app' (Figure 4.). On the resulting page, navigate to 'From your organization' (Figure 5.) You will be presented with a list of solutions to install into the site.

You will be presented with a list of solutions to install into the site.

Figure 4. 'Add an app'

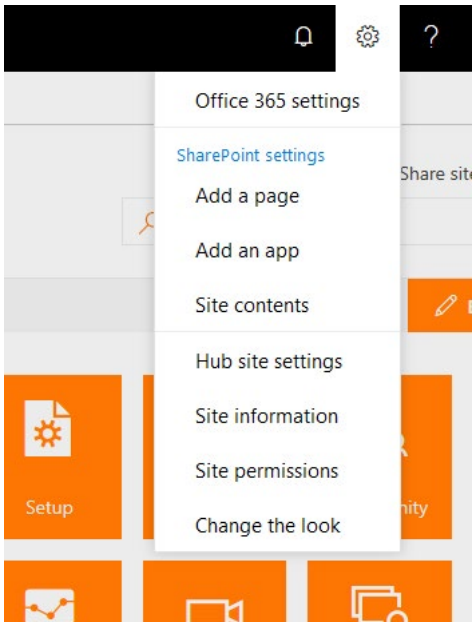
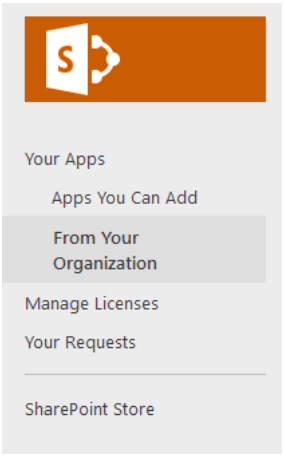


Figure 5. 'From your organization'



4.3.2. Installing a solution with PnP PowerShell

To upgrade a solution using PnP PowerShell you first need to retrieve the Id of the installed app:

```
Connect-PnPPOnline https://<yourtenant>.sharepoint.com/sites/yoursite
Get-PnPApp
```

This will return a list of available and installed solutions. Copy the id of the solution you want to install to your clipboard.

To upgrade a solution:

```
Install-PnPApp -Identity <id of the solution>
```

4.3.3. Installing a solution with the Office 365 CLI

First make sure to connect to site collection where you want to install the solution and list the apps available so you can copy the id:

```
spo connect https://<yourtenant>.sharepoint.com/sites/yoursite
spo app list
```

You will be presented with a list of all available solutions. Copy the id of the solution to your clipboard and execute:

```
spo app install --id <id of the solution> --siteUrl https://<yourtenant>.
sharepoint.com /sites/yoursite
```

4.4. Upgrading solutions

As your solution evolves, you will likely need to upgrade it to use fixes and additional features implemented by developers.

4.4.1. Manually upgrading solutions

You can upgrade solutions manually by navigating to the app catalog and simply uploading (or drag and dropping) the upgraded solution.

With PowerShell you can use the `Add-PnPApp` cmdlet in combination with the `-Overwrite` parameter. In the Office 365 CLI you can use the `spo app add` command with the `--overwrite` switch.

Now that the solution has been updated in the app catalog you will have to update the actual solution in the site too. This is not done automatically, with the exception of globally deployed solutions. If a solution can be globally deployed depends on how a developer built the solution. A globally deployed solution is available to every site collection and is automatically updated if the app has been updated in the app catalog.

4.4.2. Upgrading a solution with PnP PowerShell

To upgrade a solution using PnP PowerShell you first need to retrieve the Id of the installed app:

```
Connect-PnPPOnline https://<yourtenant>.sharepoint.com/sites/yoursite
Get-PnPApp
```

This will return a list of available and installed solutions. Notice that if a solution is installed, the InstalledVersion column will be populated with a value. Copy the id of the solution you want to upgrade to your clipboard.

To upgrade a solution:

```
Update-PnPApp -Identity <id of the solution>
```

4.4.3. Upgrading a solution with the Office 365 CLI

First make sure to connect to site collection where you want to upgrade the solution and list the apps available so you can copy the id:

```
spo connect https://<yourtenant>.sharepoint.com/sites/yoursite
spo app list
```

You will be presented with a list of all available solutions. Copy the id of the solution to your clipboard and execute:

```
spo app upgrade --id <id of the solution> --siteUrl https://<yourte-
nant>.sharepoint.com /sites/yoursite
```

4.5. Uninstalling solutions

If your organization doesn't use the particular solution anymore, it's recommended to remove it from your tenant. The first step is to uninstall it from sites where it has been installed.

4.5.1. Manually uninstalling solutions

Uninstalling a solution means that you remove a solution from a site collection, but keep it available in the app catalog.

4.5.2. Uninstalling a solution with PnP PowerShell

To uninstall a solution using PnP PowerShell you first need to retrieve the Id of the installed app:

```
Connect-PnPPOnline https://<yourtenant>.sharepoint.com/sites/yoursite
Get-PnPApp
```

This will return a list of available and installed solutions. Notice that if a solution is installed, the InstalledVersion column will be populated with a value. Copy the id of the solution you want to uninstall to your clipboard.

To uninstall a solution:

```
Uninstall-PnPApp -Identity <id of the solution>
```

4.5.3. Uninstalling a solution with the Office 365 CLI

First make sure to connect to site collection where you want to uninstall the solution and list the apps available so you can copy the id:

```
spo connect https://<yourtenant>.sharepoint.com/sites/yoursite
spo app list
```

You will be presented with a list of all available solutions. Copy the id of the solution to your clipboard and execute:

```
spo app uninstall --id <id of the solution> --siteUrl https://<yourte-
nant>.sharepoint.com /sites/yoursite
```

4.6. Removing solutions

When a solution is not used anymore and has been removed from all sites, it can be safely removed from the app catalog.

4.6.1. Manually removing solutions

The procedure to remove a solution is relatively similar to upgrading a solution. If you want to do it manually, navigate to the applicable app catalog, locate the solution and delete the entry from the app catalog.

4.6.2. Removing a solution with PnP PowerShell

To remove a solution using PnP PowerShell you first need to retrieve the Id of the installed solution:

```
Connect-PnPPOnline https://<yourtenant>.sharepoint.com/sites/yoursite
Get-PnPApp
```

This will return a list of available solutions. Copy the id of the solution you want to remove to your clipboard.

To remove the solution:

```
Remove-PnPApp -Identity <id of your solution>
```

4.6.3. Removing a solution with the Office 365 CLI

First make sure to connect to site collection where you want to remove the solution and list the apps available so you can copy the id:

```
spo connect https://<yourtenant>.sharepoint.com/sites/yoursite
spo app list
```

You will be presented with a list of all available solutions. Copy the `id` of the solution to your clipboard and execute:

```
spo app remove --id <id of the solution>
```

5. What you should know before deploying the solution

Because SharePoint Framework solutions have unrestricted access to the information and APIs in your tenant on behalf of the current user, there are some important questions that you should answer before deploying the solution in your tenant.

5.1. What does the solution truly do

Along with the solution package you might have received documentation describing the solution. But have you verified if the documentation is complete and up-to-date and if there have been last-minute changes to the solution that aren't included in the documentation?

Depending on what the solution does, your organization might have a different way of handling it. Maybe it doesn't even allow certain solutions to be used in specific sites. Just because the documentation says the solution does or doesn't do something doesn't mean it's true.

5.2. Where is the solution hosted

SharePoint Framework solutions consist of two main parts: a manifest with the solution's metadata and JavaScript files containing the solution's code.

Developers building SharePoint Framework solutions can choose whether they want to include scripts inside the solution package or not. If they do, when you deploy the package to your app catalog, these scripts will be copied to a document library in your tenant. But if they don't, then it means that the scripts are hosted elsewhere and that leads to a number of additional questions.

5.2.1. Who owns the hosting location

Is the server, where the script files are hosted, owned by your organization or is it owned by developers? Is the continuity of this hosting location guaranteed or will you wake up one day to an email saying that one or more web parts are broken?

5.2.2. Under which circumstances could the source code change

SharePoint Framework scripts have unrestricted access to your tenant. This is why you need to be able to ensure their integrity. Under what circumstances could the source code of the deployed files change? Who has access to the files and could the files be changed without an official process? Is this in agreement with your organizational policies?

5.2.3. What is the SLA of the hosting location

If SharePoint Framework solutions are hosted outside of SharePoint, then it's possible that the hosting location doesn't meet the same SLA as SharePoint. This could lead to planned outage of some components on your intranet during working hours.

5.3. Can the solution be deployed globally

Based on the contents of the solution, developers can decide to support tenant-wide deployment or not. When a SharePoint Framework solution supports tenant-wide deployment, it can be deployed to all sites at once. This alleviates you from having to manually deploy the solution to all sites and keeping track of where the solution is deployed. If the solution needs to provision some resources that it needs to function properly, like lists or content types, you might have to deploy it on each site separately.

5.4. Does it require access to APIs

When building SharePoint Framework solutions, developers can specify that the solution requires access to resources secured with Azure Active Directory. These resources can be Microsoft APIs, like the Microsoft Graph, or enterprise applications owned by your organization. When deploying a package that requests access to APIs, you will be prompted to visit the SharePoint administration page and either approve or reject these requests. There are two caveats that you should consider.

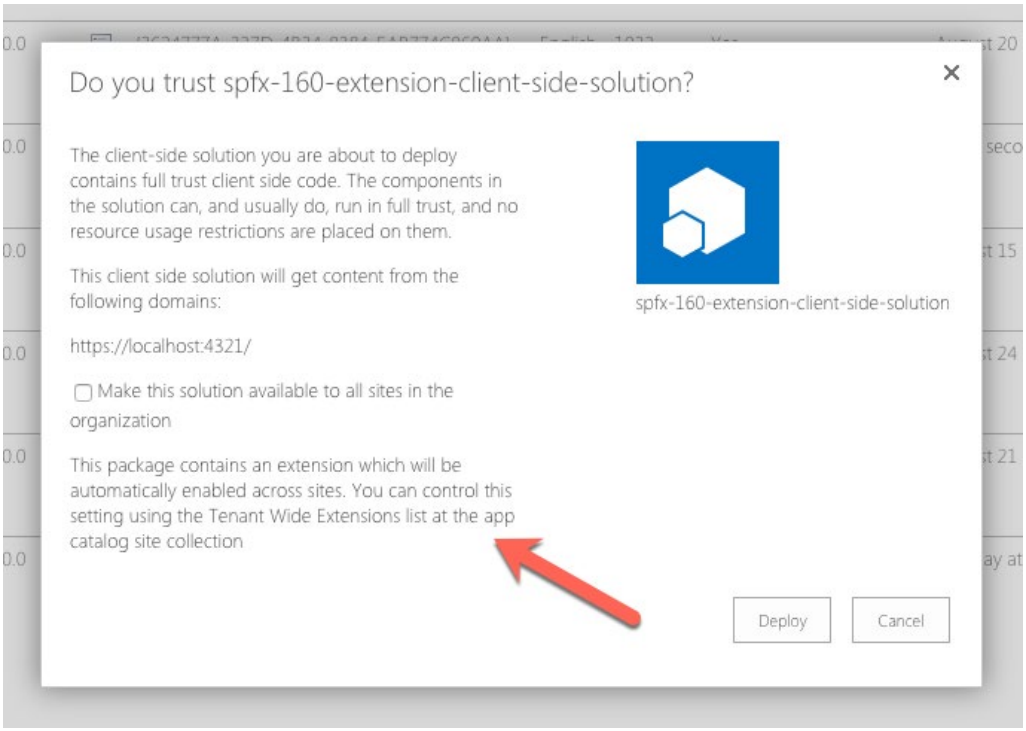
First of all, just because a solution doesn't request permissions to access any APIs, it doesn't mean it's not using them. Solution's code and the permissions it requests are two separate things not related to each other.

Once you approve a permission requested by a solution, that permission is granted to all scripts in your tenant: both scripts deployed through SharePoint Framework packages as well as all arbitrary scripts that your users might embed on their pages. This is why you should be very careful what APIs you allow to be used in your tenant.

5.5. Does it activate extensions globally

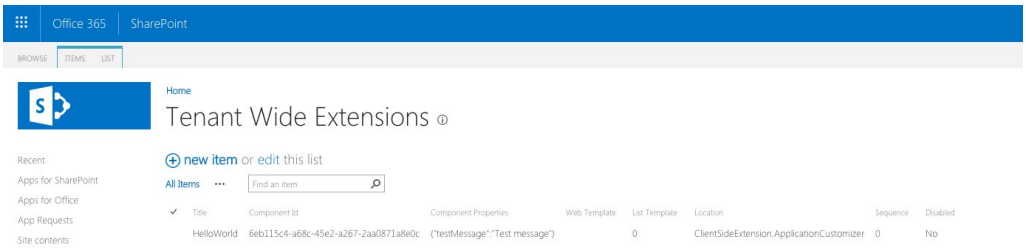
Starting from SharePoint Framework version 1.6.0, developers have the ability to automatically activate SharePoint Framework extensions. They can decide if actions should be activated globally on all sites, or only on a specific type of site, like Team Sites. When you add a SharePoint Framework solution package with one or more globally activated extension, you will get a warning similar to the Figure 6.

FIgure 6. Warning about the SharePoint Framework solution package containing globally activated extension



You can review which extensions are globally deployed in your tenant by navigating to the Tenant Wide Extensions list in the tenant app catalog (Figure 7).

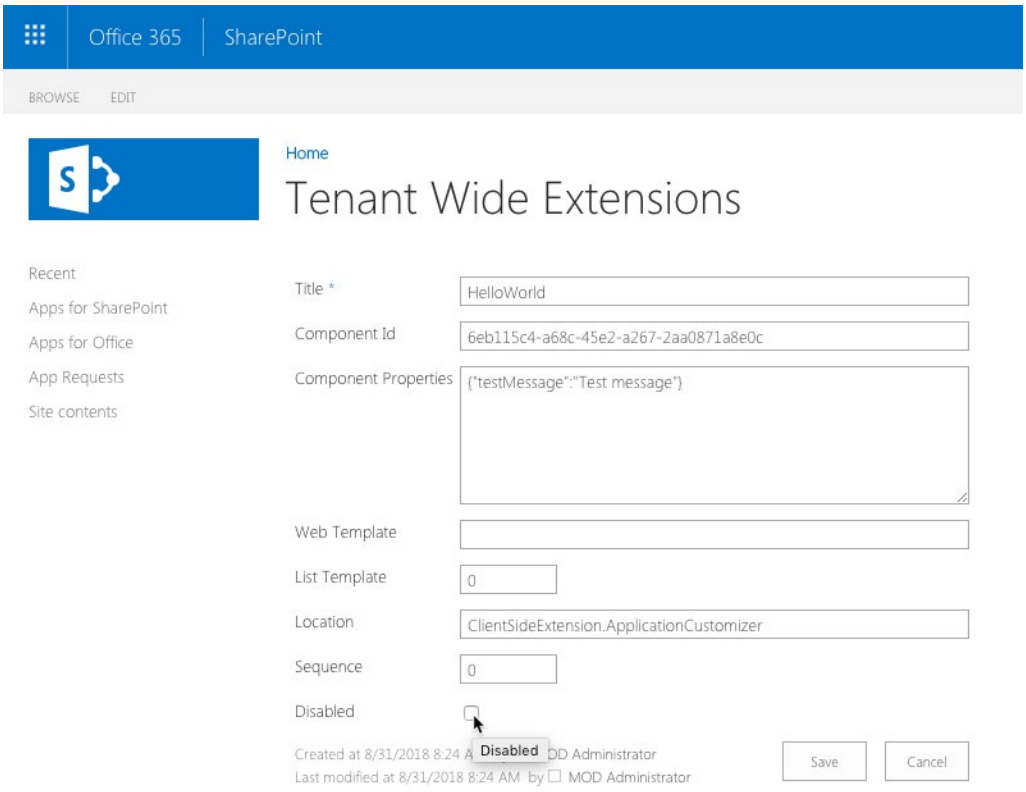
FIgure 7. List of globally activated extensions



On one hand, being able to globally activate extensions is convenient, because it allows you to consistently activate extensions on all sites across your tenant. But if you're not careful, you might activate an extension that was intended for a specific site, or a set of sites across all sites in the whole tenant. Unfortunately, it is developers who decide which extensions should be globally activated when building the package. The good news is, that after deploying the package, you can adjust how extensions are activated through the Tenant Wide Extensions list.(Figure 8).

FIgure 8.

Control how extensions are activated through the Tenant Wide Extensions list



5.6. What JavaScript libraries does it use

There are literally thousands of JavaScript libraries available on the internet that developers can use in their solutions. While using an existing library enhances developer productivity, it comes with caveats and risks for the organization using the solution.

5.6.1. Is there anyone in the organization who knows this library

When developers use a specific JavaScript library to build their solution on, it introduces a huge dependency for your organization. What if that developer moves on? Is there someone else in the organization capable of fixing bugs or applying changes without having to rebuild the whole solution? How likely is the library to exist in 6 months' time or a year? How active is the community supporting it and how well do they respond to any issues? If your organization doesn't standardize which libraries developers should use, you could face costly problems.

5.6.2. Where does the solution load its JavaScript libraries from

Similarly to the scripts that make up a SharePoint Framework solution, all JavaScript libraries used by that solution are loaded and executed as a part of the page. These libraries also have unrestricted access to all information in your tenant and all APIs exposed through Office 365. Can you trust the location hosting these scripts to ensure their integrity? What is the possibility of someone modifying these scripts and adding malicious code that could access the data in your tenant? Also, if an issue is discovered, is there someone you can contact? How likely are they to fix the issue?

5.6.3. What is the SLA of this hosting location

If any of the JavaScript libraries used by the SharePoint Framework solution fails to load, that whole solution will likely fail to load as well. If developers have choosen to load JavaScript from a location not owned by your organization, you should verify that the SLA at least meets the SLA of SharePoint.

5.6.4. What is the license of this library

JavaScript libraries are often distributed under a license. Some, don't specify a license explicitly. Do you know what the licenses are of the JavaScript libraries used in the solution and is the software distributed under these licenses approved to be used by your legal department?

5.6.5. Are there any known vulnerabilities

Despite the best intentions, regularly, vulnerabilities are being found in JavaScript libraries. The severity and impact of these vulnerabilities varies and often each reported vulnerability contains a recommendation whether you should update to a different version or take some other corrective action. There are a number of services on the internet tracking vulnerabilities in JavaScript libraries. Have you checked that the libraries used in the solution are up to date and don't contain any known vulnerabilities?

5.7. Does the solution allow loading arbitrary scripts

Developers building SharePoint Framework solutions can allow users to specify a piece of JavaScript to embed in the solution. Whether it's for extensibility or configuration purposes, if a solution allows end-users to include an arbitrary piece of JavaScript, it should be treated with additional caution, similarly to the Content- and Script Editor Web Parts in the classic SharePoint UI. Scripts that can be embedded by end-users circumvent organizational governance and security policies.

If a SharePoint Framework web part allows end-users to load arbitrary scripts, it should clearly identify it by having the **`requiresCustomScript`** flag in the manifest set to **`true`**. Such web part will only be allowed to be used on sites that allow using custom script (where the **`no-script`** setting is disabled). The important detail is that the value of the **`requiresCustomScript`** property is not binding. A web part could have the **`requiresCustomScript`** property set to **`false`** and yet allow end-users to embed arbitrary scripts.

6. Verify SharePoint Framework solutions with Rencore

It's hard to verify if the SharePoint solution that you have is safe to deploy to production or not. It's one thing to check that it works as expected and another to ensure that it meets your organization's policies, and that it doesn't contain any vulnerabilities or malicious code. To do that, you need specialized tooling that will help you examine and understand the package's contents.

A survey we recently conducted among SharePoint administrators shows that only a third of them have the means and expertise to examine a solution's package contents before deploying it to production. And the risk is real, because 30% of the administrators we interviewed stated at least 50% of the issues reported are related to poorly implemented SharePoint applications. And the number of issues related to SharePoint applications in some organizations goes up as high as 100%.

A poorly implemented SharePoint application can affect the availability of your SharePoint environment, but it also exposes your organization to data loss risks which can cost your organization as much as €5 million not to mention the loss of reputation.

Rencore specializes in analysis of SharePoint applications. Using our solutions, you can easily examine the contents of SharePoint applications and ensure that they meet your organization's policies before deploying them to production. Using our vast database of over 1000 rules, you will be able to assess your application's security, performance, supportability and overall quality.

SharePoint allows users to build applications directly in the production environment without packaging or deploying them, for example by using the Script Editor Web Part. Using Rencore, you will be able to discover which applications are running in your SharePoint tenant and if there are any potential issues that expose your organization to data - or productivity loss.

Rencore integrates with a variety of tools used in DevOps processes, from Microsoft Visual Studio Team System to PowerShell, to help you secure your SharePoint environment and find issues as soon as possible. As a result, your organization saves time and money and can focus on its core business instead of responding to issues.

Visit https://rencore.com/try today and see for yourself how it would help your organization securely extend SharePoint to its needs.

About authors

Erwin van Hunen

Erwin van Hunen is a Microsoft MVP and Microsoft Certified Master working at Rencore. A product owner for transformation tooling, Erwin assists customers with migrating from on-premises solutions to the new add-in model for SharePoint. He is a core member of the SharePoint Patterns and Practices team and a regular speaker at SharePoint conferences across the globe.

Email: erwin@rencore.com

Microsoft Accreditation: Office Apps & Services MVP

Twitter: @erwinvanhunen

Waldek Mastykarz

Waldek Mastykarz is a Product Owner at Rencore, where he designs and builds market-leading solutions that help SharePoint and Office 365 customers to minimize risks from customizing SharePoint. As a member of the SharePoint PnP core team, Waldek helps SharePoint developers get the most out of SharePoint's extensibility capabilities.

For his contributions to the technical Microsoft communities as a blogger, speaker, and developer, Waldek is a 10-time recipient of the Microsoft MVP award.

Email: waldek@rencore.com

Microsoft Accreditation: Office Development MVP

Blog: https://blog.mastykarz.nl/

Twitter: @waldekm

About Rencore

Rencore is the leading provider of software protecting organizations against customization risks in Microsoft SharePoint and Office 365.

Their proven suite of products detects and resolves potential security issues and compliance risks caused by custom coded solutions, third-party components, and citizen developers.

Rencore's de-facto standard tools are addressing three key areas: Risk Prevention, Transformation and Governance. Rencore's leading-edge software is trusted by more than 500 organizations worldwide and is used by Microsoft among other major enterprises.

Address

Bayerstr. 71-73

80335 Munich, Germany

Email

support@rencore.com

sales@rencore.com

Phone

US: +1 (855) 9-769-769

UK: +44 (800) 808 5121

DE: +49 (89) 143 79 43 91

Fax: +49 (89) 215 41 69-99

rencore.com

Your organizaiton will benefit from the SharePoint Framework, you only need to make sure you have a governance plan in place. This is where Rencore will help you succeed.

rencore.com