Create a Client-side Web Part

Nicolae Caprarescu
FULL-STACK ENGINEER

www.properjava.com

Module Overview

Configure the Application Property Pane

Add content to show the current user

Package and deploy to the app catalog

Install to a SharePoint site

Globomantics Requirements

Globomantics wants to have an application that will show useful information about the current user

It will show the Staff number along with their Line Manager, colleagues and direct reports

Globomantics also wants the application to be available to all of their sites

Property Fields

- Button
- Checkbox
- Choice group
- Dropdown
- Link
- Slider
- Textbox
- Multi-line Textbox
- Toggle
- Custom

Demo

Configure the application's Property Pane by adding a new property

Context Properties

- Web title
- Full URL
- Relative URL
- User sign-in name

Demo

Add content to the web part to show some information about the current user

```
<p class="${ styles.description }">Name:
${escape(this.context.pageContext.user.displayName)}</p>
```

Add the user name

Here we will add the user's display name to the webpart using one of the context properties shown before.

Microsoft Graph

- Is a powerful API exposing information about the tenant
- Controlled by permissions that must be requested and approved by admin
- Can fetch users and their relationships
- Can be easily integrated with custom applications
- Is available to everyone with a Microsoft 365 tenant

```
"webApiPermissionRequests": [
 {
  "resource": "Microsoft Graph",
  "scope": "User.Read.All"
 }
]
```

- These are the permissions required by the web part to query user information using the Microsoft Graph API
- These are placed in the **package-solution.json** file inside the **config** folder

```
.rowTable {
  display: flex;
 }


.columnTable3 {
  flex: 33%;
  padding: 5px;
}
```

- These are two pieces of css that will help us to render the web part and the information about the current user in a 3 column format

```
public render(): void {
  this.context.msGraphClientFactory
  .getClient()
  .then((client: MSGraphClient): void => {
    // get information about the current user from the Microsoft Graph
    client
    .api('/me')
    .get((error, userProfile: any, rawResponse?: any) => {
      this.domElement.innerHTML = `
     <div class="${ styles.globoSkeleton}">
       <div class="${ styles.container}">
         <div class="${ styles.row}">
           <span class="${ styles.title}">Welcome ${escape(this.context.pageCo
ntext.user.displayName)}!</span>
           <div class="${ styles.subTitle}" id="spUserContainer"></div>
           <div class="${ styles.rowTable}">
             <div class="${ styles.columnTable3}">
               <h2>Manager</h2>
               <div id="spManager"></div>
             </div>
             <div class="${ styles.columnTable3}">
               <h2>Colleagues</h2>
               <div id="spColleagues"></div>
             </div>
             <div class="${ styles.columnTable3}">
               <h2>Direct Reports</h2>
               <div id="spReports"></div>
             </div>
           </div>
         </div>
       </div>
     </div>`;

  });
}
```

- We replace the existing **render** method with this new version.
- In this you can see the containers where we will be adding the information about the user's manager, colleagues and direct reports

```
private _renderJobTitle(userProfile: MicrosoftGraph.User): void {
  const spUserContainer: Element = this.domElement.querySelector('#spUserContainer');
  let html: string = spUserContainer.innerHTML;
  html += `<p>${escape(userProfile.jobTitle)}</p>`;
  spUserContainer.innerHTML = html;
}
```

Render the User's Job Title

This piece of code will use the return value from the first query to Microsoft Graph and extract the user's job title from their profile.

```
private _renderEmployeeId(client: MSGraphClient): void {
  client
  .api('/me/employeeId/$value')
  .get((error, employeeId: any, rawResponse?: any) => {
   const spUserContainer: Element = this.domElement.querySelector('#spUserContainer');
   spUserContainer.innerHTML += `<p>${escape(employeeId)} </p>`;
  });
}
```

Render the Employee Id

All of the user's in the tenant have had their Employee Id field populated with values. This method will fetch the Employee Id from the user profile using the /me/employeeId endpoint in a new Graph query.

```
private _renderDirectReports(client: MSGraphClient): void {
  client
  .api('/me/directReports')
  .get((error, directReports: any, rawResponse?: any) => {
   const spContainer: Element = this.domElement.querySelector('#spReports');
   let html: string = spUserContainer.innerHTML;
   directReports.value.forEach((directReport: MicrosoftGraph.User) => {
    html += `<p class="${styles.description}"> ${escape(directReport.displayName)} </p>`;
   });
   spContainer.innerHTML = html;
  });
}
```

Render the direct reports

This method will call the Graph API to extract the user's direct reports using the /me/directReports end point.

```
private _renderManagerAndColleagues(client: MSGraphClient, userProfile: MicrosoftGraph.User): void {
 client
  .api('/me/manager')
  .get((error, manager: MicrosoftGraph.User, rawResponse?: any) => {
   const spUserContainer: Element = this.domElement.querySelector('#spManager');
   let html: string = spUserContainer.innerHTML;
   if (manager != null) {
    html += `<p class="${styles.description}"> ${escape(manager.displayName)} </p>`;
    spUserContainer.innerHTML  = html;
   }
   this._renderColleagues(client, userProfile, manager.id);
  });
}
```

Render Manager and Colleagues

This method will extract the manger id using the /me/manager endpoint in a graph query. It then renders the manager into the correct container and executes the renderColleagues method with the manager id as one of the parameters.

```
private _renderColleagues(client: MSGraphClient, userProfile: MicrosoftGraph.User, managerId: string): void {
  client
   .api(`/users/${managerId}/directReports`)
   .get((error, directReports: any, rawResponse?: any) => {
     const spColleagueContainer: Element = this.domElement.querySelector('#spColleagues');
     let html: string = spColleagueContainer.innerHTML;
     directReports.value.forEach((directReport: MicrosoftGraph.User) => {
       if (directReport.id != userProfile.id) {
         html += `<p class="${styles.description}">${escape(directReport.displayName)}</p>`;
       }
     });
     spColleagueContainer.innerHTML = html;
   });
}
```

Render Colleagues

Using the manager id from the calling method, this will execute another graph query endpoint /users/{managerId}/directReports to extract the user's colleagues. As the list of direct reports will contain the current user as well, we need to exclude them from the list rendered.

Demo

Package the web part

Deploy to the app catalog

Demo

Install the application on a SharePoint site

SharePoint Facts

Launched in 2001

Over 100 million users worldwide

Used by over 50% of Fortune 500 companies

https://microsoft.com/sharepoint

Module Summary

- Configured the Application Property Pane
- Added content to show the current user on the web part
- Packaged and deployed to the app catalog site
- Installed the application to a SharePoint site