LUIS MAÑEZ

Microsoft® Most Valuable Professional

KIOTA ❤ SPFx

Using a KIOTA generated client in an SPFx webpart

#SharePoint Framework community call 15th Dec 2022

About Me

Luis Mañez

Chief Architect @ ClearPeople

M365 Development MVP. More than 20 years of experience working with Microsoft Technologies. MS Certified specialist in Developing Azure and SharePoint MCPD. Passionate about technology, blogger, technical writer and MS community active member.

@luismanez

Community Call Presenter
M365 PLATFORM COMMUNITY
2022

MVP
Microsoft
Most Valuable Professional

NEST (from Swahili)

What is Kiota?

Kiota is a command line **tool** for generating [illegible] **OpenAPI** described API you are interested in. The goal is to eliminate the need to ta[illegible]nt API SDK for every API that you need to call. Kiota API clients provide a strongl[illegible]eatures you expect from a high-quality API SDK, but without having to learn a[illegible]

Fast and scalable source code generator to simplify calling HTTP APIs

Provide support for a wide range of languages: C#, Java, Typescript, PHP, Ruby, Go, Python, Swift, Shell

[illegible]ry

Lightweight, easy to install command line tool

Enable generation of code for only a specified subset of an OpenAPI description

PUBLIC PREVIEW

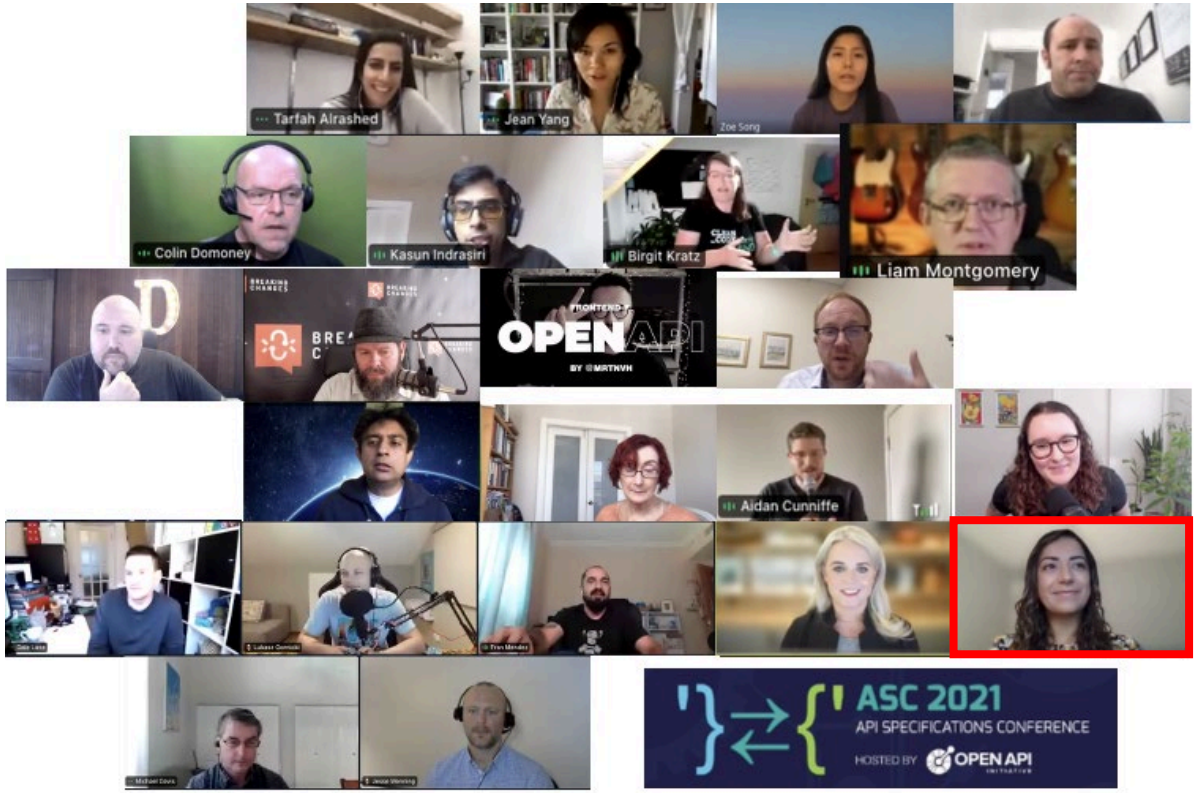
https://microsoft.github.io/kiota/

```
kiota generate --openapi openapi.json --language csharp --output client
```

```
kiota generate (--openapi | -d) <path>
      (--language | -l) <language>
      [(--output | -o) <path>]
      [(--class-name | -c) <name>]
      [(--namespace-name | -n) <name>]
      [(--log-level | --ll) <level>]
      [--backing-store | -b]
      [--additional-data | --ad]
      [(--serializer | -s) <classes>]
      [(--deserializer | --ds) <classes>]
      [--clean-output | --co]
      [--clear-cache | --cc]
      [(--structured-mime-types | -m) <mime-types>]
      [(--include-path | -i) <glob pattern>]
      [(--exclude-path | -e) <glob pattern>]
```

OPENAPI INITIATIVE

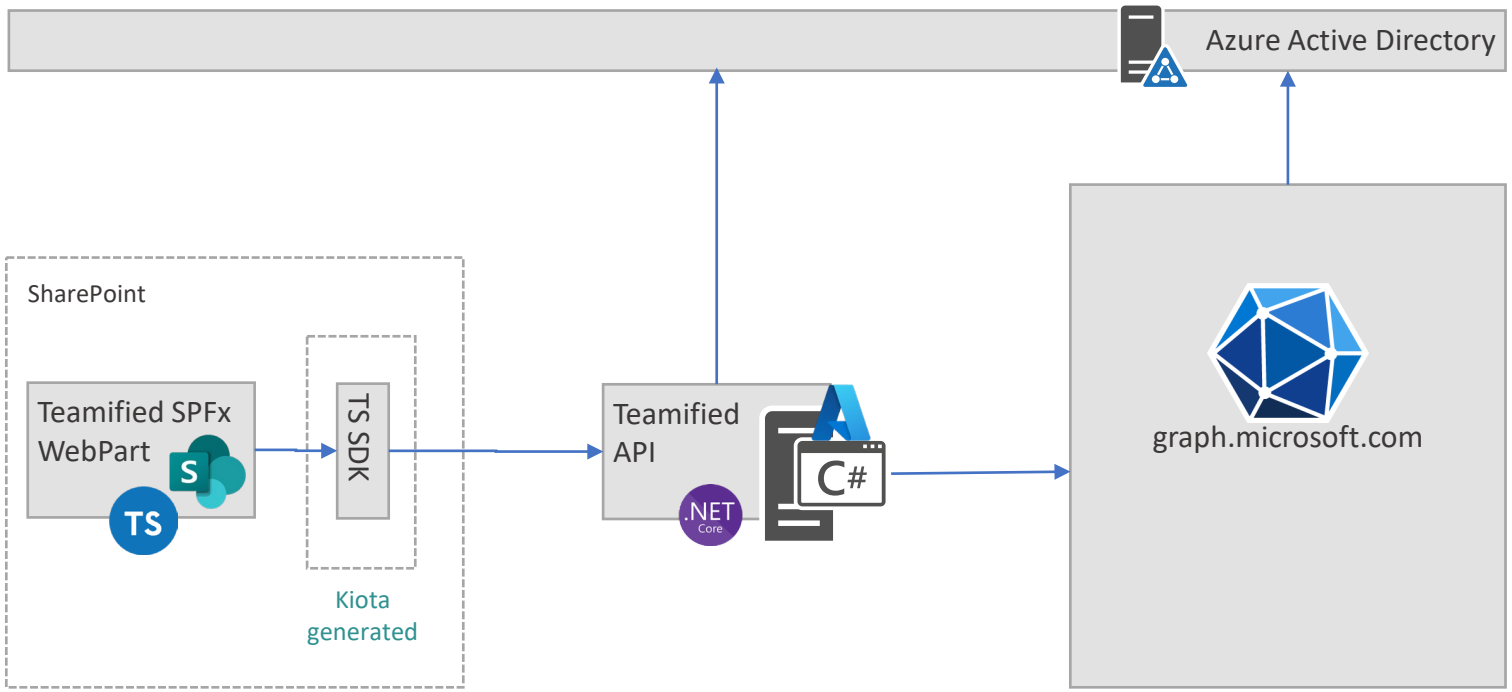
The OpenAPI Initiative (OAI) was created by a consortium of forward-looking industry experts who recognize the immense value of standardizing on how APIs are described. As an open governance structure under the Linux Foundation, the OAI is focused on creating, evolving and promoting a vendor neutral description format. The OpenAPI Specification was originally based on the Swagger Specification, donated by SmartBear Software.



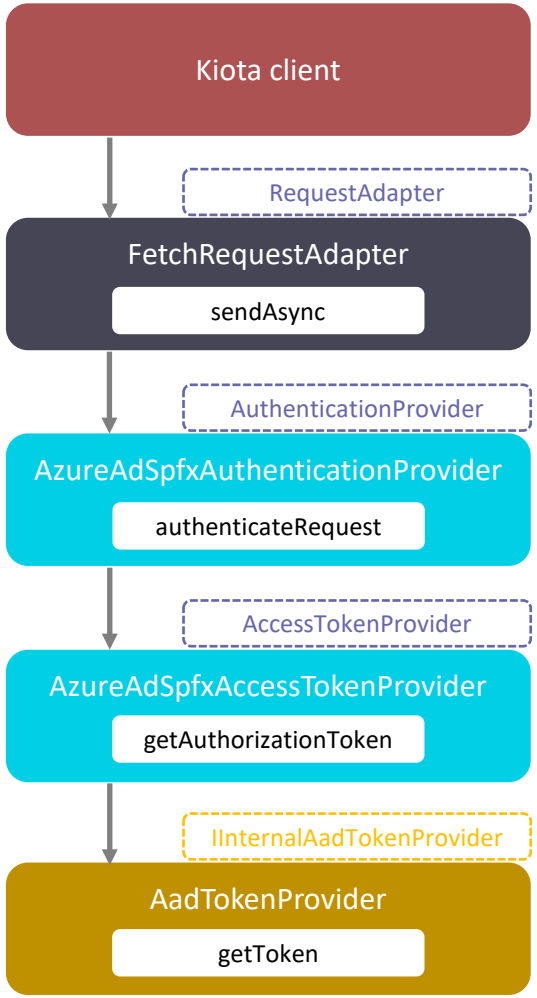

```yaml
openapi: 3.0.3
info:
  title: Microsoft Graph get user API
  version: 1.0.0
servers:
  - url: https://graph.microsoft.com/v1.0/
paths:
  /me:
    get:
      responses:
        200:
          description: Success!
          content:
            application/json:
              schema:
                $ref: "#/components/schemas/microsoft.graph.user"
components:
  schemas:
    microsoft.graph.user:
      type: object
      properties:
        id:
          type: string
        displayName:
          type: string
```

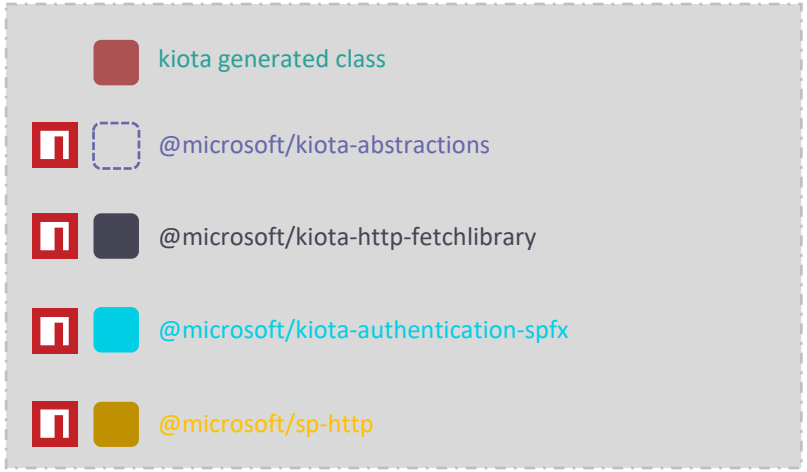
https://oai.github.io/Documentation/

Teamified Demo Architecture

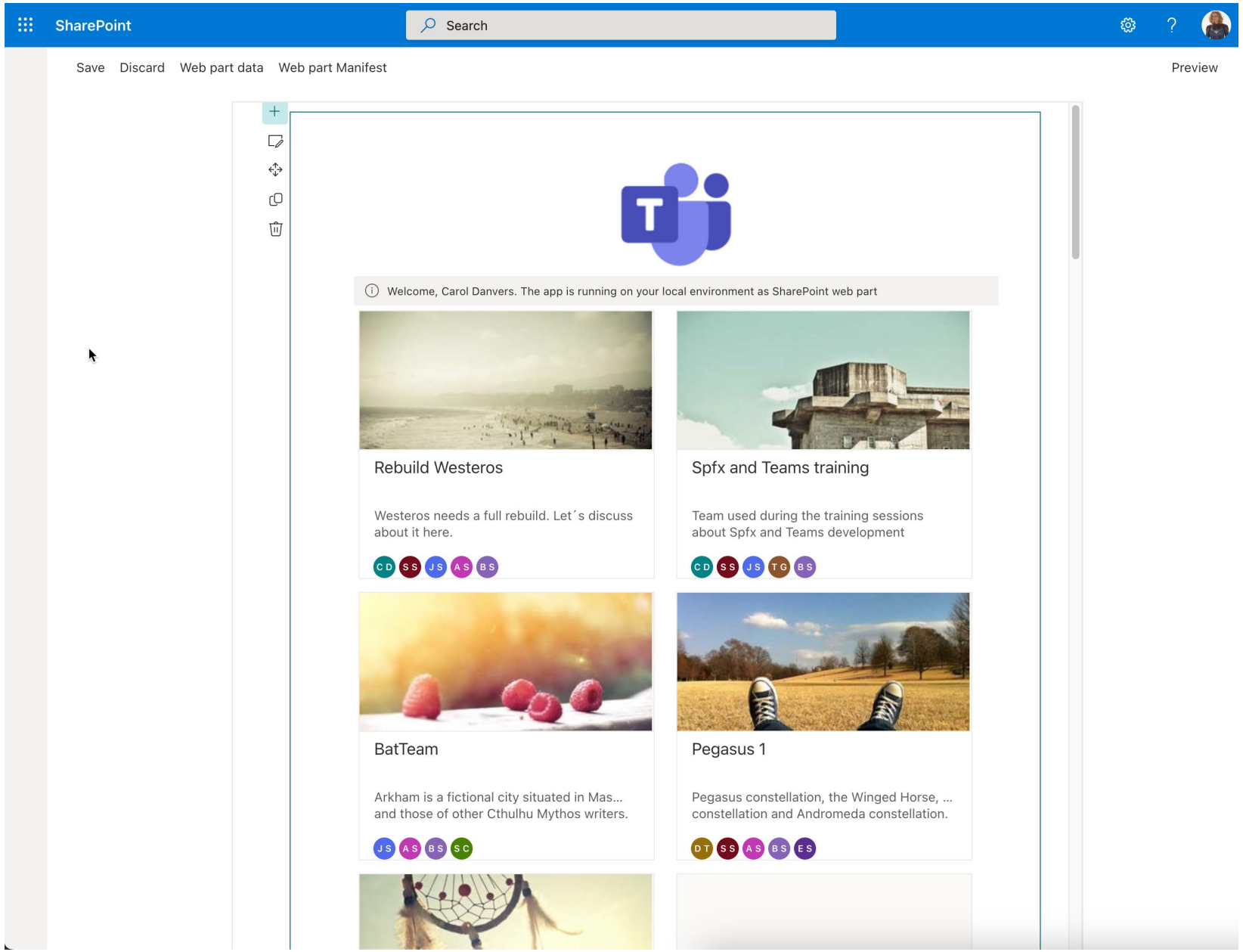


Kiota client authentication in SPFx





Let´s Dive In!


SharePoint
Search
Save
Discard
Web part data
Web part Manifest
Preview
Welcome, Carol Danvers. The app is running on your local environment as SharePoint web part
Rebuild Westeros
Westeros needs a full rebuild. Let´s discuss about it here.
CD SS JS AS BS
Spfx and Teams training
Team used during the training sessions about Spfx and Teams development
CD SS JS TG BS
BatTeam
Arkham is a fictional city situated in Mas... and those of other Cthulhu Mythos writers.
JS AS BS SC
Pegasus 1
Pegasus constellation, the Winged Horse, ... constellation and Andromeda constellation.
DT SS AS BS ES

```
kiota generate --openapi "openapi.json" --language typescript --namespace-name
TeamifiedSdk --class-name TeamifiedApiClient --output kiota-generated --log-level debug
```

```
~/g/sp-dev-fx-webparts/s/react-kiota-custom-api-client/TeamifiedApi  react-kiota-custom-api-client !1
❯ kiota generate --openapi "openapi.json" --language typescript --namespace-name TeamifiedSdk --class-name TeamifiedAp
iClient --output kiota-generated --log-level debug
dbug: Kiota.Builder.KiotaBuilder[0]
      step 1 - reading the stream - took 00:00:00.0010420
dbug: Kiota.Builder.KiotaBuilder[0]
      step 2 - parsing URI patterns - took 00:00:00.0001782
dbug: Kiota.Builder.KiotaBuilder[0]
      step 3 - parsing the document - took 00:00:00.0664240
dbug: Kiota.Builder.KiotaBuilder[0]
      step 4 - checking whether the output should be updated - took 00:00:00.0172652
dbug: Kiota.Builder.KiotaBuilder[0]
      step 5 - filtering API paths with patterns - took 00:00:00.0001118
```

```
kiota-generated
  models
    channel.ts                                U
    createChannelFromDiscriminator...         U
    createIdentityPrincipalFromDiscri...      U
    createPingFromDiscriminatorValu...        U
    createProvisionTeamCommandFr...           U
    createTeamFromDiscriminatorVal...         U
    identityPrincipal.ts                      U
    index.ts                                  U
    ping.ts                                   U
    provisionTeamCommand.ts                   U
    team.ts                                   U
  ping
    pingRequestBuilder.ts                     U
    pingRequestBuilderGetRequestC...          U
  teams
    item
    teamsRequestBuilder.ts                    U
    teamsRequestBuilderGetRequest...          U
    teamsRequestBuilderPostRequest...         U
  kiota-lock.json                             U
  teamifiedApiClient.ts                       U
```

```
kiota info -d /Users/luisman/github/sp-dev-fx-webparts/samples/react-kiota-custom-api-
client/TeamifiedApi/openapi.json -l TypeScript
```

```
pi-client/TeamifiedApi/openapi.json
~/g/sp-dev-fx-webparts/s/react-ki/TeamifiedApi  react-kiota-custom-api-client !1 ?2
❯ kiota info -d /Users/luisman/github/sp-dev-fx-webparts/samples/react-kiota-custom-api-client/TeamifiedApi/openapi.js
on -l TypeScript
The language TypeScript is currently in Experimental maturity level.
After generating code for this language, you need to install the following packages:
npm install @microsoft/kiota-abstractions@1.0.0-preview.6 -S
npm install @microsoft/kiota-http-fetchlibrary@1.0.0-preview.8 -S
npm install @microsoft/kiota-serialization-json@1.0.0-preview.8 -S
npm install @microsoft/kiota-authentication-azure@1.0.0-preview.5 -S
npm install @microsoft/kiota-serialization-text@1.0.0-preview.7 -S
~/g/sp-dev-fx-webparts/s/react-ki/TeamifiedApi  react-kiota-custom-api-client !1 ?3
❯
```

```
builder.Services.AddAuthentication(JwtBearerDefaults.AuthenticationScheme)
        .AddMicrosoftIdentityWebApi(builder.Configuration)
          .EnableTokenAcquisitionToCallDownstreamApi(options => builder.Configuration.Bind("AzureAd", options))
            .AddMicrosoftGraph(builder.Configuration.GetSection("MicrosoftGraph"))
            .AddInMemoryTokenCaches(); // Advice: Use Distributed TokenCache (redis, sql...)

builder.Services.AddAuthorization(cfg => {
    cfg.FallbackPolicy = new AuthorizationPolicyBuilder()
        .RequireAuthenticatedUser()
        .Build();
});
```

react-kiota-custom-api-client/TeamifiedApi/Program.cs

```
endpoints.MapGet("/teams", ListTeams.Handle)
    .Produces<IEnumerable<Team>>(200)
    .WithName("ListTeams")
    .WithTags("TeamsModule");

endpoints.MapGet("/teams/{id:guid}", GetTeam.Handle)
    .Produces<Team>(200)
    .WithName("GetTeam")
    .WithTags("TeamsModule");

endpoints.MapPost("/teams", ProvisionTeam.Handle)
    .Produces<string>(202)
    .WithName("ProvisionTeam")
    .WithTags("TeamsModule");
```

react-kiota-custom-api-client/TeamifiedApi/Teams/TeamsModule.cs

```
public async Task<IEnumerable<Models.Team>> ListTeams()
{
    var teamsCollection = await _graphServiceClient.Groups
        .Request()
        .Filter("resourceProvisioningOptions/Any(x:x eq 'Team')")
        .Expand("members($select=id,displayName,userPrincipalName,jobTitle,mail)")
        .Select(g => new { g.Id, g.DisplayName, g.Description, g.Members })
        .GetAsync();

    var teams = teamsCollection.CurrentPage;

    var result = teams.Select(t => Models.Team.MapFromGraphGroup(t));

    return result;
}
```

react-kiota-custom-api-client/TeamifiedApi/Teams/Queries/ListTeams/ListTeamsQuery.cs

```
public async Task<string> ProvisionTeam(Models.Team team)
{
    var currentUserId = _context.HttpContext.User.FindFirst("http://schemas.microsoft.com/identity/claims/objectidentifier").Value;

    var newTeam = new Team()
    {
        DisplayName = team.DisplayName,
        Description = team.Description,
        AdditionalData = new Dictionary<string, object>()
            {
                {"template@odata.bind", "https://graph.microsoft.com/v1.0/teamsTemplates('standard')"}
            },
        Members = new TeamMembersCollectionPage()
        {
            new AadUserConversationMember
            {
                Roles = new List<string>()
                {
                    "owner"
                },
                AdditionalData = new Dictionary<string, object>()
                {
                    {"user@odata.bind", $"https://graph.microsoft.com/v1.0/users/{currentUserId}"}
                }
            }
        }, Channels = new TeamChannelsCollectionPage
        {
            new Channel
            {
                DisplayName = "KickOff Channel",
                IsFavoriteByDefault = true,
                Description = "As per company policy, place here data related with the Kickoff of the Team"
            }
        }
    };

    var result = await _graphServiceClient.Teams
                .Request()
                .AddResponseAsync(newTeam);

    if (result.HttpHeaders.TryGetValues("Location", out var locationValues))
    {
        return locationValues?.First();
    }

    return "Something went wrong. Location not found";
}
```

react-kiota-custom-api-client/TeamifiedApi/Teams/Commands/ProvisionTeam/ProvisionTeamCommand.cs

```
public render(): void {
  const element: React.ReactElement<ITeamsListProps> = React.createElement(
    TeamsList,
    {
      description: this.properties.description,
      isDarkTheme: this._isDarkTheme,
      environmentMessage: this._environmentMessage,
      hasTeamsContext: !!this.context.sdks.microsoftTeams,
      userDisplayName: this.context.pageContext.user.displayName,
      aadTokenProviderFactory: this.context.aadTokenProviderFactory
    }
  );

  ReactDom.render(element, this.domElement);
}
```

react-kiota-custom-api-client/teamified-client/src/webparts/teamsList/TeamsListWebPart.ts

```
public componentDidMount(): void {
  this.props.aadTokenProviderFactory.getTokenProvider()
    .then((tokenProvider: AadTokenProvider): void => {

      const authProvider =
        new AzureAdSpfxAuthenticationProvider(
          tokenProvider,
          this.azureAdApplicationIdUri,
          new Set<string>([
            this.apiHost
          ]));

      const adapter = new FetchRequestAdapter(authProvider);
      adapter.baseUrl = `https://${this.apiHost}`;

      const teamifiedClient = new TeamifiedApiClient(adapter);

      teamifiedClient.teams.get().then(teams => {
        console.log(teams);
        this.setState({
          teams: teams
        });
      })
      .catch(e => {console.log(e)});
    })
    .catch(e => {console.log(e)});
}
```

react-kiota-custom-api-client/teamified-client/src/webparts/teamsList/components/TeamsList.tsx

```
const personas: IFacepilePersona[] = t.members.map(m => {
  const nameSplit: string[] = m.displayName.split(' ');
  const firstNameInitial: string = nameSplit[0].substring(0, 1).toUpperCase();
  const lastNameInitial: string = nameSplit[1] ? nameSplit[1].substring(0, 1).toUpperCase() : '';
  const persona: IFacepilePersona = {
    personaName: m.displayName,
    imageInitials: `${firstNameInitial} ${lastNameInitial}`
  }
  return persona;
});

return (
    <DocumentCard key={t.id} type={DocumentCardType.normal}>
      <DocumentCardDetails>
        <DocumentCardPreview {...previewProps} />
        <DocumentCardTitle title={t.displayName} />
        <DocumentCardTitle title={t.description} showAsSecondaryTitle shouldTruncate />
        <Facepile personas={personas} personaSize={PersonaSize.size24} className={styles.facepile} />
      </DocumentCardDetails>
    </DocumentCard>);
```

react-kiota-custom-api-client/teamified-client/src/webparts/teamsList/components/TeamsList.tsx

https://github.com/pnp/sp-dev-fx-webparts/tree/main/samples/react-kiota-custom-api-client

References

- This sample
  - https://github.com/pnp/sp-dev-fx-webparts/tree/main/samples/react-kiota-custom-api-client
- Kiota official docs
  - https://microsoft.github.io/kiota/
- Kiota GitHub repo
  - https://github.com/microsoft/kiota
- Kiota TypeScript repo (includes SPFx auth package)
  - https://github.com/microsoft/kiota-typescript
- My blog posts about Kiota:
  - https://www.clearpeople.com/blog/microsoft-kiota-tool-to-generate-atlas-api-sdks
  - https://www.clearpeople.com/blog/howto-kiota-client-api-sharepoint-framework-solution

Thanks!

#sharingiscaring

Microsoft® Most Valuable Professional

@luismanez