MCP Server URL

Use this URL to access the Adaptive Planning MCP Server:

```
https://<MCP Hostname>/<Planning Tenant Name>-<Planning Tenant Environment>/mcp/?appName=<Planning Instance Code>
```

Example:

```
https://api-us.prvw.adaptiveplanning.com/GREENCO-PREVIEW/mcp/?appName=GREENCO_PLN
```

MCP Hostnames by Region

- US: api-us.prvw.adaptiveplanning.com
- Canada: api-ca.prvw.adaptiveplanning.com
- Australia: api-au.prvw.adaptiveplanning.com
- Europe: api-eu.prvw.adaptiveplanning.com
- India: api-in.prvw.adaptiveplanning.com
- Japan: api-jp.prvw.adaptiveplanning.com

Adaptive Planning Tenant Name, Environment, and Instance Code

In your Adaptive Planning instance, you can find these details:

- Environment and tenant name in the top banner. Example: PREVIEW - GREENCO.
- Instance code from the **Administration**, **General Setup** page. You can see the instance code in the **Code** field.

For Standalone Adaptive Planning instances

1. Retrieve OAuth server metadata including all supported endpoints, scopes, and algorithms.

   Request:

```
curl -s https://<MCP
Hostname>/<tenant>-<environment>/.well-known/oauth-authorizatio
n-server | jq .
```

Example:

```
curl -s
https://api-us.prvw.adaptiveplanning.com/GREENCO-PREVIEW/.well-
known/oauth-authorization-server | jq .
```

The Response includes issuer, authorization_endpoint, token_endpoint, jwks_uri, scopes_supported, id_token_signing_alg_values_supported (RS512).

2. Register an API client by calling the REST API using admin basic auth with username and password.

Request:

```
curl -s -X POST \
https://api.prvw.adaptiveplanning.com/api/rest/security/v1/defa
ult/clients \
  -u <admin_username>:<admin_password> \
  -H "Content-Type: application/json" \
  -d '{
    "clientName": "<client_name>",
    "clientType": "CONFIDENTIAL",
    "grantTypes": ["authorization_code", "refresh_token"],
    "redirectUris": ["<redirect_url>"],
    "scopes": ["MCP-Basic"],
    "requireConsent": true,
    "requirePkce": true
  }' | jq .
```

The response includes a JSON object containing clientId and clientSecret.

```
{
  "clientId": "<client_id>",
  "clientSecret": "<client_secret>",
  "clientName": "<client_name>"
}
```

Note: To find the API url for other regions, see [Adaptive Planning Authentication URLs and IP Addresses].

3. Generate PKCE Parameters:
    a. Generate a code verifier:

```
# Generate code_verifier (43-128 URL-safe base64 chars)
CODE_VERIFIER=$(openssl rand -base64 32 | tr -d '=' | tr
'+/' '-_')
echo "code_verifier: $CODE_VERIFIER"
```

    b. Generate a code challenge:

```
# code_challenge = BASE64URL(SHA256(code_verifier))
CODE_CHALLENGE=$(echo -n "$CODE_VERIFIER" | openssl dgst
-sha256 -binary | base64 | tr -d '=' | tr '+/' '-_')
echo "code_challenge: $CODE_CHALLENGE"
```

4. Call the /authorize endpoint in your browser to get auth_code:

```
https://<MCP Hostname>/<tenant>-<environment>/oauth2/authorize
  ?response_type=code
  &client_id=<client_id>
  &redirect_uri=<redirect_url>
  &scope=MCP-Basic
  &state=random_state_value
  &code_challenge=<code_challenge>
  &code_challenge_method=S256
```

   After sign-in and consent, the browser redirects to this URL:

```
<redirect_url>?code=auth_code&state=....
```

5. Use the auth_code and code_verifier to obtain access and refresh tokens. Authenticate with client Basic auth (clientId:clientSecret).

   Request:

```
curl -s -X POST \
https://<MCP Hostname>/<tenant>-<environment>/oauth2/token\
  -u <client_id>:<client_secret> \
  -d "grant_type=authorization_code" \
  -d "redirect_uri=<redirect_url>" \
  -d "code_verifier=<code_verifier>" \
 -d "code=<auth_code>" | jq .
```

   Response:

```
{
  "access_token": "<access_token>",
  "refresh_token": "<refresh_token>",
```

```
  "scope": "MCP-Basic",
  "token_type": "Bearer",
  "expires_in": 3599
}
```

The access token is a signed JWT (RS512). Decode the token at jwt.io to verify claims: tenant, env, instance_code, sub (userguid), tokenType=APToken.

6. Use the refresh token to obtain a new access token without re-authenticating the user.

   Request:

   ```
   curl -s -X POST \
     https://<MCP Hostname>/<tenant>-<environment>/oauth2/token \
     -u <client_id>:<client_secret> \
     -d "grant_type=refresh_token" \
     -d "refresh_token=<refresh_token>" | jq .
   ```

   The response includes a new access_token and a rotated refresh_token. The old refresh token is immediately invalidated.

For Workday-Enabled Adaptive Planning instances (Authorization Code Grant Method)

1. Register a Workday API client using the authorization code grant method. See Register API Clients.
2. Retrieve the Workday Access Token:
   a. Sign in as an end user to a Workday HCM or Financial Management tenant.
   b. Call this URL:

      ```
      https://<host or gateway>/wday/authgwy/<workday tenantname>/authorize?response_type=code&client_id={client_id}
      ```

      Response:

      ```
      https://<redirect uri>?code=<authorization_code>
      ```

      Example request:

      ```
      https://impl.workday.com/greenco/authorize?response_type=code&client_id=ZDlkZDBlZTMtOsdGIwZC00NzRjLThmOWehTg4MGNjY2FmNDc2
      ```

      Example response:

```
https://www.greenco.com/?code=1lm6nyoeyrqptsm5d8rnd9djp
```

3. To get the access token, call the token endpoint with client ID, client secret, and authorization code:

   Request:

```
curl --request POST \
  --url https://<host or gateway>/ccx/oauth2/<workday
tenantname>/token \
  --header 'authorization: Basic
(base64encode<clientID>:<clientSecret>)' \
  --header 'content-type: application/x-www-form-urlencoded' \
  --data grant_type=authorization_code \
  --data code=<authorization_code>
```

   Response:

```
{
  "access_token": "<access token>",
  "token_type": "Bearer",
  "refresh_token": "<refresh token>"
```

4. Retrieve the AdaptiveAPIAccessToken by calling the Workday endpoint using the Workday access token. Use the prior access token value as the bearer token in the Auth header for this GET.

   Note: Skip this step if Unified Provisioning and Authentication System (UPAS) is enabled.

   Example Access Token:

```
https://{host or
     gateway}/ccx/api/planning/v1/{tenant}/adaptiveAPIAccessTok
     en
```

   Example request:

```
curl --request GET \
  --url
https://wd2-impl.workday.com/ccx/api/planning/v1/greenco/adapti
veAPIAccessToken \
  --header 'authorization: Bearer <access token>' \
  --header 'content-type: application/x-www-form-urlencoded'
```

For Workday-Enabled Adaptive Planning instances (JWT Bearer Grant Method)

MCP authentication is similar to API authentication. Create an integration system user (ISU) specifically for MCP requests and map it to an Adaptive Planning user account. For step-by-step instructions, see Make Adaptive Planning API Requests with Workday Credentials.

Note: Skip this step if Unified Provisioning and Authentication System (UPAS) is enabled.

Call the MCP Server

After generating the token using one of the documented methods, complete these steps to connect to the Adaptive Planning MCP Server:

1. Configure your AI agent to call the MCP Server.
2. Call the MCP server url and pass the access token as bearer token in the authorization header to initialize the MCP session. See MCP Server URL.
3. Pass the returned mcp-session-id header on subsequent tool calls.

Audit Log

You can access the audit logs for the tool calls using the audits REST API. You need the *Activity Auditing* permission for calling the API.
GET:
Request:
`https://api.adaptiveplanning.com/api/rest/audits/v1/default/activities`