Oracle Utilities Opower GraphQL API User Guide

F96049-05
June 2025

Oracle Utilities Opower GraphQL API User Guide,

F96049-05

Contents

1

Oracle Utilities Opower GraphQL API User Guide

Welcome to the *Opower GraphQL API User Guide*. Use this information to learn about the capabilities of the Oracle Utilities Opower GraphQL API offering and how to utilize the GraphQL API resources. Have a question? Contact Your Delivery Team or visit My Oracle Support.

The Oracle Utilities Opower Integration Hub includes access to a set of GraphQL APIs. These APIs can be used to create personalized and seamless web or mobile experiences that increase customer satisfaction and drive customer action at scale.

Working with GraphQL APIs requires retrieving an API access token, and using the proper client-specific base URL.

Quick Links

2

Requirements and Limitations

The following data requirements and limitations apply to all utilities using the GraphQL APIs.

Utility Data Requirements and Limitations

The following data requirements and limitations apply to all utilities using the GraphQL APIs.

- **Data Integration**: You must complete data integration to use Oracle Utilities Opower data and insights with Opower Integration Hub. Data must be sent to Oracle Utilities in the right schema and according to the established data specifications. Your Oracle Utilities Delivery Team will work with you to identify which data specifications are applicable to your situation.
  - **AMI Insights:** AMI insights are available if AMI data is provided to Oracle Utilities in the required format. For more information, refer to the Oracle Utilities Opower Data Transfer Standards.
  - **Rates**: Utility rates must be modeled by Oracle Utilities for rates data and insights to be available. For more information on the Rates Engagement Cloud Service, refer to the Rates Engagement topic of the Oracle Utilities *Opower* Digital Self Service - Energy Management Cloud Services *Overview*.
- **Authentication**: A utility must configure their instance of Oracle Identity and Access Management to allow for authentication and single sign-on (SSO) integration. This includes enabling the use of OpenID Connect Authorization Code Flow. Information on implementing Authorization Code Flow with Oracle Identity and Access Management and request examples can be found in the Identity and Access Management documentation.
- **GraphQL Introspection and Query Testing**: Reviewing the GraphQL schema and building queries can be accomplished on a test instance that provides access to a GraphQL IDE. General access and schema introspection is available to all users. Third-party partners must submit a service request through My Oracle Support for access to execute queries from this testing location. Oracle Utilities responds to these service requests by providing applicable account information, credentials, and access tokens for this test environment. Be aware that access tokens expire and no longer provide query execution access 60 minutes after their creation.
- **API Rate Limiting**: The GraphQL API endpoint is limited to 175 queries per second and 250,000 API calls per day for each application instance.
- **Data Querying and Availability**: Be aware of the following data querying and availability details.
  - **Billing Data**:
    - Billing data can be returned for up to 25 bills in a single request.
    - Data is available for the last two years preceding the time of the API request.
    - If the active utility account is associated with multiple service points of the same fuel type, billed usage per service point is returned.
  - **Interval Usage**:
    - Daily data can be returned for up to 45 days in a single request.

* Days are defined from midnight to midnight in the local timezone of the site or service point.
* If the active utility account is associated with multiple service points of the same fuel type, usage data per service point is returned.

3

Accessing the GraphQL APIs Using Utility-Scoped Tokens

The Oracle Utilities GraphQL APIs require proper authentication through access tokens and base URL construction to gain access and use of the APIs. A user can obtain a utility-scoped access token that allows retrieving user data from the GraphQL APIs for all accounts within a utility.

Retrieving Utility-Scoped Access Tokens

The GraphQL APIs support the OAuth 2.0 standard for authentication. OAuth 2.0 is an authorization framework that enables an application or a service to obtain limited access to a protected HTTP resource. In OAuth, the applications are called clients, and clients access protected resources by presenting an access token to the HTTP resource.

OAuth 2.0 is typically used in the "three-legged" fashion, in which an end-user authorizes a third-party to access their data. However, the Oracle Utilities GraphQL API uses "two-legged" authorization, which allows a trusted developer (for example, a utility company) to access all data Oracle Utilities provides.

Obtaining Credentials

Provide the following information, using https://support.oracle.com, to your Oracle Utilities Customer Success Representative:

- Contact name
- Contact email
- The specific utility code to access if multiple operating companies exist

Oracle Utilities provides you with the following keys:

- `client_id`
- `client_secret`
- `grant_type`

For example, a client's contact information is sent through https://support.oracle.com to Oracle Utilities:

- **Name**: Integration Team
- **Email**: IntegrationTeam@utilityX.com
- **Utility Access**: UtilityX

The client receives Oracle Utilities information through a secure email:

- `client_id: 019ebf3c-bc9c-41fa-87aa-09b1ecc86012`
- `client_secret: bdf46a70-c553-44bd-846e-7bd4f9d98690`

- `grant_type: client_credentials`

It is recommend that you retain all information provided to obtain credentials as it is required in the case of a forgotten `client_id`.

Securing your Credentials and Requesting New Credentials

Keys can be used to access utility customer data, so they must be protected and stored securely. Keys must have limited distribution, they should not be shared among developers, and they must only be used by authorized utility personnel. Do not use keys or tokens directly in a browser or mobile application. To make data from Oracle Utilities GraphQL APIs available to customers, you must create a server-side API gateway to accept customer requests for data using their session token. Your server-side API gateway must then make a request to the Oracle Utilities GraphQL APIs using your utility's OAuth token.

If you suspect that a key has been compromised or its security can no longer be assured, contact Oracle Utilities using https://support.oracle.com with the `client_id`. A new `client_secret` is issued.

If credentials are forgotten, you can request new credentials from Oracle Utilities. If the `client_secret` is forgotten, you must send a communication using https://support.oracle.com with your `client_id`. Oracle Utilities will issue a new `client_secret`. If the `client_id` is forgotten, you must send a communication using https://support.oracle.com with the information provided when first obtaining credentials, and Oracle Utilities issues a new `client_id` and `client_secret`.

Obtaining a Utility-Scoped Access Token

Before calling other API endpoints for a given tier, a client must obtain a utility-scoped access token from the authorization-v1 endpoint (`authorization-v1/oauth2/token`) using the issued client credentials.

To obtain a client access token:

1. Use a curl command to request the utility-scoped access token. If you choose to use a different method for this type of request, be aware that the `-u` option below performs Base64 encoding on the `client_id:client_secret` value. The syntax of this command is as follows:

```
curl -XPOST host:port/apis/authorization-v1/oauth2/token -d
"grant_type=client_credentials" -u "client_id:client_secret" -H "Content-
Type: application/x-www-form-urlencoded"
```

You must replace `host`, `port`, `client_id`, and `client_secret` with the applicable values.

An example response is provided below:

```
{
  "client_id":"019ebf3c-bc9c-41fa-87aa-09b1ece86012",
   "access_token":"a3b96ccb-cff0-4c12-b693-40e7158aa8cd",
   "token_type":"BEARER",
   "scope":"",
   "expires_in":3599
}
```

The `expires_in` parameter is expressed in seconds.

2. After retrieving the utility-scoped access token, you can provide the utility-scoped access token as part of your GraphQL queries, which is described in Using the GraphiQL IDE and Using Curl. If the utility-scoped access token expires, repeat the process to obtain a valid access token.

Constructing the Base URL

A GraphiQL Integrated Development Environment (IDE) provides access for you to explore the schema and run queries using utility-scoped access tokens. You can access the IDE using the following URL structure:

- **Stage Environment**: `https://api-stage.us.opower.com/graphql`

  This environment is intended for API call and feature testing.

- **Production Environment**: `https://api.us.opower.com/graphql`

  This environment is intended to be used for updates or retrievals from your production system that directly supports your customers.

4

Accessing the GraphQL APIs Using Account-Scoped Tokens

Third-party partner applications and users can access the set of GraphQL APIs that support a utility environment. The integration from a third-party partner application to a utility's GraphQL API environment requires a connection between the application and the utility's Oracle Identity and Access Management instance through OAuth2 and OpenId Connect. A user can obtain an account-scoped access token that allows retrieving user data from the GraphQL APIs for a particular account.

> **Note:**
>
> This topic discusses a GraphQL API integration scenario between Oracle Utilities, a third-party partner, and a utility. A utility can also choose to complete the integration requirements of a third-party partner directly without involvement with a third-party partner, and in this scenario a utility would complete all third-party partner integration requirements listed below.

Retrieving Account-Scoped Access Tokens

Access to GraphQL API data and queries requires access tokens to confirm authentication. GraphQL API users are required to retrieve and extract an authorization code, and use this value along with other identifying information to receive an account-scoped access token.

Requirements to Supply to Oracle Utilities

The third-party partner is required to provide Oracle Utilities with redirect URIs, which are the URLs where the authorization codes are sent to the third-party application. A redirect URI is required for each utility a third-party partner integrates with, and each environment or tier for that utility, which commonly consists of a stage environment and production environment. This means at least two redirect URIs are commonly required for each utility that a third-party partner integrates with.

Along with redirect URIs, third-party partners must provide an explanation of the type of access required. For example, a third-party may request read style access for utility customer billing data. These requests are used to determine the required scopes to grant to the third-party partner.

Requirements Supplied to a Third-Party Partner

The following information is provided by Oracle Utilities to third party partners for each applicable utility client.

- **OpenId Connect Discovery Endpoints**: Endpoints are provided for each environment a utility has, which commonly includes a stage environment and a production environment.

The endpoints are of the format `https://idcs-[instance_id].oraclecloud.com/.well-known/openid-configuration`.

- **Client Credentials**: Each set is applicable to a given utility and the relevant environments:
  - `client_id`
  - `client_secret`
- **Scopes**: A list of scopes is provided by Oracle Utilities based on the level of access requests. Be aware that if scopes are missing this can result in errors when using the GraphQL APIs. This results in errors that list information such as

  ```
  Required scopes 'scope_name' not present in authorization context
  ```

  . If this occurs, contact Oracle Utilities to review scope definitions and confirm the required scopes are included in the application redirects.

Configuring the Authorization Flow to Obtain Account-Scoped Access Tokens

A third-party partner must configure their application as part of the authorization flow to obtain the required account-scoped access tokens. The authorization flow is described below, and clarifies the steps in the flow that require third-party partner configuration support.

1. A user clicks a link in the third-party web application, requesting access to protected resources.
2. The application redirects the browser to the Oracle Identity and Access Management authorization endpoint with a user login request.

   The third-party partner must configure this application redirect. The request URL contains query parameters that indicate the scope of access being requested and is in the following format:

   ```
   GET https://idcs-[instance_id].oraclecloud.com/oauth2/v1/authorize?
   client_id=[client_id]&response_type=code&redirect_uri=[redirect_uri]&scope
   =[scopes]&state=[random_string]
   ```

   Where:

   - `[instance_id]` can be retrieved from one of the OpenId Connect Discovery Endpoints provided by Oracle Utilities.
   - `[client_id]` is provided by Oracle Utilities.
   - `[redirect_uri]` is the redirect URI that is provided by the third-party partner to Oracle Utilities.
   - `[scopes]` is the full list of scopes to include in the request, and these scope names are provided by Oracle Utilities. In addition to the scopes provided by Oracle Utilities, it is required to include the scope `openid` in the list of scopes.
   - `[random_string]` is a randomized string provided by the third-party partner.
   - An example request is provided below:

     ```
     https://idcs-45fa2455571f40e087f43f850d8f1337.identity.oraclecloud.com/
     oauth2/v1/authorize?
     ```

```
response_type=code&client_id=Opower_SSO_util_stage_testpartner_APPID&sco
pe=openid+opower_consumer_util_testpartner_stageperm.func.coreEntities.u
tilityIds.read&redirect_uri=https://testpartner.com/auth/
redirect&state=fd7f976d-24bc-4ae4-8c9b-891814df170e
```

3. If the user is not already logged in, Oracle Identity and Access Management challenges the user to authenticate, and verifies the user's credentials.
4. If the login is successful, Oracle Identity and Access Management redirects the user's browser back to the application with an authorization code. An example of this response is provided below:

   ```
   https://testpartner.com/auth/redirect?code=891814df170e
   ```

5. The third-party partner application must be configured to extract the authorization code and makes a request to an identity domain to exchange the authorization code for an access token.

   ```
   curl -i -u [client_id]:[client_secret]
   -H 'Content-Type: application/x-www-form-urlencoded;charset=UTF-8'
   --request POST https://idcs-[instance_id].oraclecloud.com/oauth2/v1/token
   -d 'grant_type=authorization_code&code=[authorization_code]'
   ```

   Where:

   - `client_id` and `client_secret` are provided by Oracle Utilities, and this operation performs Base64 encoding on the values.
   - `[instance_id]` can be retrieved from one of the OpenId Connect Discovery Endpoints provided by Oracle Utilities.
   - `[authorization_code]` is the authorization code extracted from the response provided by Oracle Identity and Access Management.
   - An example request is provided below:

     ```
     curl -i -u 'Authorization: Basic
     a2ZmamRhbGtma2xkYXZrbGRha2xkYWexy2RhaXVmICBhZGlvZmogYWxkZmpsamUgbCBxb2U'
     -H 'Content-Type: application/x-www-form-urlencoded;charset=UTF-8'
     --request POST https://
     idcs-45fa2455571f40e087f43f850d8f1337.identity.oraclecloud.com/
     oauth2/v1/token
     -d 'grant_type=authorization_code&code=891814df170e'
     ```

6. Oracle Identity and Access Management validates the grant and user data associated with the third-party partner application request to the identity domain.
7. An account-scoped access token, an identity token, and optionally a refresh token are returned. The account-scoped access token contains information about which data the application can request on behalf of the user. The application can use this token when requesting APIs on behalf of the user. These API calls can be determined and constructed through query introspection and testing as described in Building Queries and Exploring Data.

   The third-party partner application can further use the identity token to initiate the Single Logout flow with Oracle Identity and Access Management if required.

Constructing the Base URL

A GraphiQL Integrated Development Environment (IDE) provides access for you to explore the schema and run queries using account-scoped access tokens. You can access the IDE using the following URL structure:

- **Stage Environment**: `https://api-stage.us.opower.com/[utilityCode]/graphql`

  This environment is intended for API call and feature testing.

- **Production Environment**: `https://api.us.opower.com/[utilityCode]/graphql`

  This environment is intended to be used for updates or retrievals from your production system that directly supports your customers.

Where:

- `utilityCode` is a three or four letter code for a utility. Contact Oracle Utilities if you are unsure of the utility code for the utility you are integrating with.

For example, a utility with the utility code of `abc` can access their GraphQL IDE at `https://api.us.opower.com/abc/graphql`. Additionally, a demonstration area accessible for all users uses the utility code of `util` and can be accessed at `https://api.us.opower.com/util/graphql`.

5

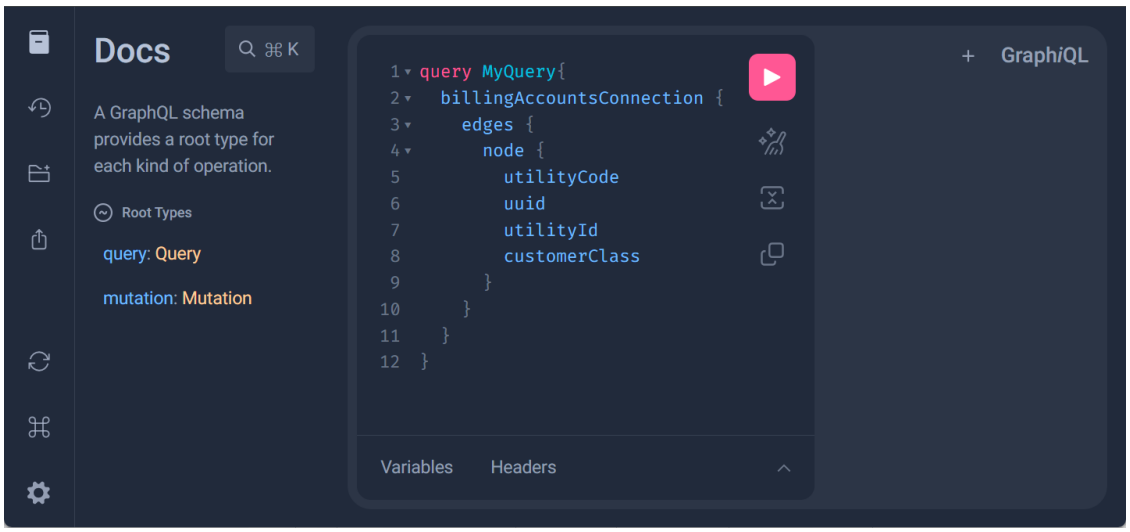
Building Queries and Exploring Data

A demonstration area can be accessed at https://api.us.opower.com/util/graphql for users who have retrieved the required credentials and account-scoped access tokens as described in Requirements and Limitations. With Access to this GraphQL Integrated Development Environment (IDE), you can begin to build your queries or mutations, and explore the available data. Along with using the IDE, you can also use curl to query the GraphQL environment.

> **Note:**
>
> For more information on the GraphQL standard, refer to https://graphql.org

Using the GraphiQL IDE

GraphiQL is the IDE available for reviewing and discovering the GraphQL APIs. For information on the user interface, the link in the top-right corner of the interface redirects to the applicable Github repository that includes documentation resources on GraphiQL.



Reviewing the Schema

From the GraphiQL IDE at https://api.us.opower.com/util/graphql, you can select to **Show Documentation Explorer** to review the schema definition. The Documentation Explorer allows you to traverse through the schema. A search field is also available to search for schema definitions of interest.

> **Note:**
>
> Schema definitions can include links to Oracle and third-party resources. Be aware that navigating to these resources can redirect your active window away from the GraphiQL IDE. You can use various methods to open these resources is separate browser windows and tabs to avoid this redirection.

Building a Query

The GraphiQL IDE allows you to easily build a query and expand on the query to explore available data. You can select to **Show GraphiQL Explorer** to display the Explorer area. This area allows you to browse through the schema and select elements to add to your query.

For example, since all queries use the `billingAccountsConnection` entry point, you can select it to include it in a new query tab. This query creation can continue by Reviewing the Schema to learn about subfields to include in your query, and typing in the new elements or selecting them from the Explorer.

As you continue to build your query, the IDE validates the syntax of your query. As you complete query definitions you can confirm that your query is valid by ensuring there are no syntax errors. The GraphQL protocol uses the concepts of edges and nodes to structure the queries. Refer to third-party GraphQL documentation if you are unfamiliar with query structure and design in GraphQL.

Building a Mutation

While queries in GraphQL allow for retrieving data, mutations provide the capability of modifying data. Be aware of the following GraphQL standards related to mutations:

- Mutations and queries cannot be included in a single request. Each style of request must be done separately.
- The order of your mutations impacts the processing of the mutation. While queries are executed in parallel, mutations are executed in series to avoid update race conditions.

You can build your mutation query by Reviewing the Schema to learn about subfields to include in your mutation, and typing in the new elements or selecting them from the Explorer. The IDE validates the syntax of your query. As you complete mutation definitions you can confirm that your mutation is valid by ensuring there are no syntax errors. Refer to third-party GraphQL documentation if you are unfamiliar with mutation structure and design in GraphQL.

Running a Query or Mutation

As part of running a query or mutation, you must supply the authorization context, which is accomplished through an access token which can be done using the following methods.

Running a Query or Mutation Using Account-Scoped Access Tokens

Request an account-scoped access token from Oracle Utilities through a service request. After retrieving an access token, the access token must be included in the HTTP header for authorization purposes. You can include the access token in the **Headers** area when building your query, using the following syntax and replacing the access token in the `AccessToken` placeholder:

> **Note:**
>
> Be aware that access tokens expire and no longer provide access 60 minutes after their creation.

```
{
    "Authorization": "Bearer AccessToken"
}
```

When you are ready to run your query, you can select the **Execute query** option. Results are displayed directly within the IDE. After you are able to validate expected results from the query you can integrate the query into your system as required.

Running a Query or Mutation Using Utility-Scoped Access Tokens

Complete the process described in Obtaining a Utility-Scoped Access Token. Be aware that when using utility-scoped tokens, an applicable utility code and account numbers must be supplied as parameters of the query. For mutations, an applicable urn must be supplied as parameters of the mutation. A valid urn for use in mutations can be retrieved through queries on the entity to modify with a mutation. These parameters are further described in Example Queries.

Be aware that while reviewing the schema and building queries using https://api.us.opower.com/util/graphql is open to all users, running queries in this location requires account-scoped access tokens. The schema review is identical regardless of the style of access token being used. Additionally, the queries used are also identical aside from the inclusion of a utility code and account numbers, or an urn for mutations, when using utility-code access tokens. For this reason, if you plan to use utility-scoped access tokens, you can use https://api.us.opower.com/util/graphql to review the schema and build your query. To successfully run a query with utility-scoped access tokens, you must do so from https://api.us.opower.com/graphql.

Using Curl

While the GraphiQL IDE provides an easy way to browse the schema and build queries or mutations, queries or mutations can also be run using curl. This can be helpful if you are building off of a known query or mutation, or sharing a query or mutation with a collaborator to test or confirm expected results.

The basic syntax for a curl command for GraphQL is as follows:

```
curl '[baseURL]' \
-X [method] \
-H '[header]' \
-d '{[query]}'
```

Where:

- `[baseURL]` is the base URL to query the GraphQL resources. The demonstration area which supports account-scoped access tokens uses a base URL of `https://api.us.opower.com/util/graphql`. For more information on constructing the base URL, refer to:

  - **Utility-Scoped Access Token**: Refer to Constructing the Base URL.
  - **Account-Scoped Access Token**: Refer to Constructing the Base URL.
- `[method]` is the method for the query, such as POST.
- `[header]` can include required header information, such as supplying an access token. For information on retrieving this required access token:
  - **Utility-Scoped Access Token**: Complete the process described in Obtaining a Utility-Scoped Access Token. Be aware that when using utility-scoped tokens, an applicable utility code and account numbers, or an urn for mutations, must be supplied as parameters of the query. These parameters are further described in Example Queries.
  - **Account-Scoped Access Token**: Request an account-scoped access token from Oracle Utilities through a service request.
- `[query]` is the query or mutation to execute against the GraphQL environment. You can build a query or mutation from scratch or use the IDE as described in Using GraphiQL to build the query or mutation.

After you are able to validate expected results from the query you can integrate the query into your system as required.

Example Queries and Mutations

The data retrieved from queries or modified from mutations requires and is based on the supplied authorization context. Methods to supply the authorization context are described in Using the GraphiQL IDE and Using Curl above.

In addition to that authorization context, the following parameters are required when using utility-scoped access tokens:

- **Queries**:
  - **accountNumbers**: This parameter is an array of one or more account numbers to retrieve data for using the query.
  - **utilityCode**: This parameter is a three or four-character code that identifies the utility. The applicable value can be confirmed by your Delivery Team.
- **Mutations**:
  - **urn**: This parameter is a Uniform Resource Name, which designates the customer or account-premise edge. Refer to the Documentation Explorer for specific syntax required for urn values. A valid urn for use in mutations can be retrieved through queries on the entity to modify with a mutation.

Billing Account Example

Billing Account includes details related to the customer account including a list of service agreements, associated service points, and premise details.

Query Using Account-Scoped Access Tokens

```
query GetPremiseAndServiceDetails {
  billingAccountsConnection {
    edges {
      node {
        serviceAgreements {
```

```
            servicePointsConnection {
              edges {
                node {
                  premise {
                    address {
                      addressLines
                      country
                      postalCode
                    }
                  }
                }
              }
            }
            ratePlan {
              code
            }
            serviceType
          }
        }
      }
    }
  }
```

Query Using Utility-Scoped Access Tokens

Note: You must replace the `accountNumbers` value with a relevant account number for the query to provide a valid response.

```
query GetPremiseAndServiceDetails {
  billingAccountsConnection(
    accountNumbers: ["123456"]
    utilityCode: "util"
  ) {
    edges {
      node {
        serviceAgreements {
          servicePointsConnection {
            edges {
              node {
                premise {
                  address {
                    addressLines
                    country
                    postalCode
                  }
                }
              }
            }
          }
          ratePlan {
            code
          }
          serviceType
        }
      }
```

```
    }
  }
}
```

Response

The response includes data related to the service agreements, service points, rate plans, and service types.

```
{
  "data": {
    "billingAccountsConnection": {
      "edges": [
        {
          "node": {
            "serviceAgreements": [
              {
                "servicePointsConnection": {
                  "edges": [
                    {
                      "node": {
                        "premise": {
                          "address": {
                            "addressLines": [
                              "5933 Hackett BLVD"
                            ],
                            "country": "US",
                            "postalCode": "21784"
                          }
                        }
                      }
                    }
                  ]
                },
                "ratePlan": {
                  "code": "D"
                },
                "serviceType": "GAS"
              },
              {
                "servicePointsConnection": {
                  "edges": [
                    {
                      "node": {
                        "premise": {
                          "address": {
                            "addressLines": [
                              "5933 Hackett BLVD"
                            ],
                            "country": "US",
                            "postalCode": "21784"
                          }
                        }
                      }
                    }
```

```
                ]
              },
              "ratePlan": {
                "code": "R_PTR"
              },
              "serviceType": "ELECTRICITY"
            }
          ]
        }
      }
    ]
  }
 }
}
```

Bill Example

Bill data includes the customer's utility bill and includes a breakdown of bill segments and bill line items. Billing data can be returned for up to 25 bills in a single request. Data is available for the last two years preceding the time of the API request. If the active utility account is associated with multiple service points of the same fuel type, billed usage per service point is returned.

Query Using Account-Scoped Access Tokens

The following query uses the `last` pagination technique to return the last two available bills, which can also be thought of as the most recently received two bills.

```
query GetLast2Bills {
  billingAccountsConnection {
    edges {
      node {
        bills(last: 2) {
          billDate
          segments {
            lineItems {
              calculatedAmount {
                currency
                value
              }
            }
            serviceQuantities {
              serviceQuantity {
                unit
                value
              }
            }
          }
        }
      }
    }
  }
}
```

Query Using Utility-Scoped Access Tokens

The following query uses the `last` pagination technique to return the last two available bills, which can also be thought of as the most recently received two bills.

> **Note:**
>
> You must replace the `accountNumbers` value with a relevant account number for the query to provide a valid response.

```
query GetLast2Bills {
  billingAccountsConnection(
    accountNumbers: ["123456"]
    utilityCode: "util"
  ) {
    edges {
      node {
        bills(last: 2) {
          billDate
          segments {
            lineItems {
              calculatedAmount {
                currency
                value
              }
            }
            serviceQuantities {
              serviceQuantity {
                unit
                value
              }
            }
          }
        }
      }
    }
  }
}
```

Response

Data and information related to the two applicable bills are provided in the response.

```
{
  "data": {
    "billingAccountsConnection": {
      "edges": [
        {
          "node": {
            "bills": [
              {
```

```
      "billDate": 2023-03-24,
      "segments": [
        {
          "lineItems": [
            {
              "calculatedAmount": {
                "currency": "USD",
                "value": 185.77
              }
            }
          ],
          "serviceQuantities": [
            {
              "serviceQuantity": {
                "unit": "KWH",
                "value": 1150
              }
            }
          ]
        },
        {
          "lineItems": [
            {
              "calculatedAmount": {
                "currency": "USD",
                "value": 28.75
              }
            }
          ],
          "serviceQuantities": [
            {
              "serviceQuantity": {
                "unit": "TH",
                "value": 11
              }
            }
          ]
        }
      ]
    },
    {
      "billDate": 2023-02-23,
      "segments": [
        {
          "lineItems": [
            {
              "calculatedAmount": {
                "currency": "USD",
                "value": 139.84
              }
            }
          ],
          "serviceQuantities": [
            {
              "serviceQuantity": {
                "unit": "KWH",
```

```
                "value": 853
              }
            }
          ]
        },
        {
          "lineItems": [
            {
              "calculatedAmount": {
                "currency": "USD",
                "value": 31.52
              }
            }
          ],
          "serviceQuantities": [
            {
              "serviceQuantity": {
                "unit": "TH",
                "value": 14
              }
            }
          ]
        }
      ]
    }
  ]
}
```

Utility Information Example

Utility information can be returned as part of a query.

Query Using Utility-Scoped Access Tokens

The following query is a simple example that demonstrates returning utility information.

```
query AccountNumbersParameterExample {
  billingAccountsConnection(
    accountNumbers: ["zpbh7tmkkjgf"]
    utilityCode: "util"
  ) {
    edges {
      node {
        utilityId
        utilityCode
      }
    }
  }
}
```

Response

Data and information related to the utility is provided in the response.

```
{
  "data": {
    "billingAccountsConnection": {
      "edges": [
        {
          "node": {
            "utilityId": "zpbh7tmkkjgf",
            "utilityCode": "util"
          }
        }
      ]
    }
  }
}
```

Tips Examples

Energy saving tip information and actions are available through GraphQL APIs. This includes detailed information on available tips, along with the ability to modify tip status for customers through tip actions. The examples below demonstrate how to return various tip data and update tip actions.

Multiple Tips Query Using Utility-Scoped Access Tokens

The following query returns tip data and information for the first two tips related to the provided account number.

```
query getTipDetails {
  billingAccountsConnection(
        utilityCode: "util"
        accountNumbers: ["ZWzgpeKL"]
  ) {
    edges {
      node {
        premisesConnection {
          edges {
            urn
            tipsConnection (first: 2) {
              edges {
                node {
                  name
                  costCategory
                }
              }
            }
          }
        }
      }
    }
```

```
    }
}
```

Multiple Tips Response

The following response is the tip data and information for the first two tips related to the provided account number. Notice there is a name field which can be used to retrieve tip data and information on a specific tip.


```
{
  "data": {
    "billingAccountsConnection": {
      "edges": [
        {
          "node": {
            "premisesConnection": {
              "edges": [
                {
                  "urn": "urn:opower:v1:account-
premise:util:uuids:b87dcb80-34a7-11ef-ab76-020017084ca7,71208d75-34a7-11ef-
ab76-020017084hu9",
                  "tipsConnection": {
                    "edges": [
                      {
                        "node": {
                          "name":
"tip545_smb_install_programmable_thermostat",
                          "costCategory": "SMART_PURCHASE"
                        }
                      },
                      {
                        "node": {
                          "name": "tip503_smb_install_window_treatments",
                          "costCategory": "SMART_PURCHASE"
                        }
                      }
                    ]
                  }
                },
                {
                  "urn": "urn:opower:v1:account-
premise:bce1:uuids:b87dcb80-34a7-11ef-ab76-020017084ca7,7120ad1a-34a7-11ef-
ab76-020017084hu9",
                  "tipsConnection": {
                    "edges": [
                      {
                        "node": {
                          "name": "tip519_smb_buy_efficient_servers",
                          "costCategory": "GREAT_INVESTMENT"
                        }
                      },
                      {
                        "node": {
                          "name": "tip516_smb_buy_energy_star_computers",
                          "costCategory": "SMART_PURCHASE"
```

```
                        }
                      }
                    ]
                  }
                }
              ]
            }
          }
        }
      ]
    }
  }
}
```

Single Tip Query Query Using Utility-Scoped Access Tokens

The following query returns tip data and information for the specified tip related to the provided account number.

```
query TipDetails {
  billingAccountsConnection (utilityCode: "util"
  accountNumbers: ["6726308534"])  {
    edges {
      node {
        premisesConnection {
          edges {
            tipsConnection(id: "tip162_EV_rates") {
              edges {
                node {
                  name
                  costCategory
                }
              }
            }
          }
        }
      }
    }
  }
}
```

Single Tip Response

The following response is the tip data and information for the specified tip related to the provided account number.

```
{
  "data": {
    "billingAccountsConnection": {
      "edges": [
        {
          "node": {
            "premisesConnection": {
              "edges": [
```

```
                {
                  "tipsConnection": {
                    "edges": [
                      {
                        "node": {
                          "name": "tip162_EV_rates",
                          "costCategory": "FREE"
                        }
                      }
                    ]
                  }
                }
              ]
            }
          }
        }
      ]
    }
  }
}
```


Tip Mutation Using Utility-Scoped Access Tokens

The following mutation modifies the tip action state for a specific tip and a specific customer based on the supplied urn value. In the example below, the tip action is modified to the `ALREADY_DO_IT` state. The required urn values can be retrieved from queries that retrieve tip data.

```
mutation performAction{
    performTipAction(urn:  "urn:opower:v1:account-
premise:util:uuids:b87dcb80-34a7-11ef-ab76-020017084ca7,7120ad1a-34a7-11ef-
ab76-020017084ca7"
    tipId: "tip545_smb_install_programmable_thermostat",
    action: ALREADY_DO_IT){
        isDone
        isIgnored
        isSaved
        allowedActions
    }
}
```

Tip Mutation Response

The following response confirms the modifications completed for the tip action definition for the tip. In this example, modifying a tip to the `ALREADY_DO_IT` state marks the tip as done, which is confirmed with the `"isDone": true` response shown below.


```
{
  "data": {
    "performTipAction": {
      "isDone": true,
      "isIgnored": false,
      "isSaved": false,
      "allowedActions": [
```

```
        "SAVE",
        "UNDO"
      ]
    }
  }
}
```

Disaggregation Analysis Examples

Dissaggregated energy use and associated costs information can be retrieved as shown in the following examples.

Most Recent Bill Query Using Account-Scoped Access Tokens

The following query retrieves the disaggregated energy use and associated costs during the most recent bill.

```
query GetBillDisaggregation {
  billingAccountsConnection(first:10) {
    edges {
      cursor
      node {
        utilityId
        uuid
        customerClass
        bills(last:1) {
          timeInterval
          billDateTime
          consumptionCategories {
            name
            ratio
            cost {value, currency}
            usage {value, unit}
          }
        }
      }
    }
  }
}
```

Response

Disaggregated energy use and associated costs during the most recent bill are included in the response.

```
{
  "data": {
    "billingAccountsConnection": {
      "edges": [
        {
          "cursor": "ZTYyYjIwZjQtMmI5ZS0xMWU2LTlhYmQtZDJhNDJhNzZiMGQ2",
          "node": {
            "utilityId": "123456",
            "uuid": "e62b20f4-2b9e-11e6-9abd-d2a42a76b0d6",
```

```
            "customerClass": "RESIDENTIAL",
            "bills": [
              {
                "timeInterval":
"2024-10-04T00:00:00-04:00/2024-11-05T00:00:00-05:00",
                "billDateTime": "2024-11-04T00:00:00-05:00",
                "consumptionCategories": [
                  {
                    "name": "APPLIANCES",
                    "ratio": 0.17,
                    "cost": {
                      "value": 17.17,
                      "currency": "USD"
                    },
                    "usage": {
                      "value": 83,
                      "unit": null
                    }
                  },
                  {
                    "name": "ELECTRONICS",
                    "ratio": 0.18,
                    "cost": {
                      "value": 18.76,
                      "currency": "USD"
                    },
                    "usage": {
                      "value": 91,
                      "unit": null
                    }
                  },
                  {
                    "name": "LIGHTING",
                    "ratio": 0.05,
                    "cost": {
                      "value": 5.33,
                      "currency": "USD"
                    },
                    "usage": {
                      "value": 26,
                      "unit": null
                    }
                  },
                  {
                    "name": "OTHER",
                    "ratio": 0.1,
                    "cost": {
                      "value": 10.36,
                      "currency": "USD"
                    },
                    "usage": {
                      "value": 50,
                      "unit": null
                    }
                  },
                  {
```

```
                    "name": "TOTAL",
                    "ratio": 1,
                    "cost": {
                      "value": 70.01,
                      "currency": "USD"
                    },
                    "usage": {
                      "value": 263,
                      "unit": null
                    }
                  },
                  {
                    "name": "WATER_HEATING",
                    "ratio": 0.5,
                    "cost": {
                      "value": 18.39,
                      "currency": "USD"
                    },
                    "usage": {
                      "value": 13,
                      "unit": null
                    }
                  }
                ]
              }
            ]
          }
        }
      ]
    }
  }
}
```


Yearly Query Using Account-Scoped Access Tokens

The following query retrieves disaggregated consumption categories between November 8, 2023 and November 8, 2024.

```
query GetDisaggregation {
  billingAccountsConnection(first:10) {
    edges {
      node {
        utilityId
        serviceAgreementsConnection(first:10) {
          edges {
            node {
              utilityId
              serviceType
              consumptionCategories(timeInterval: "2023-11-08T00:00:00Z/
2024-11-08T00:00:00Z") {
                name
                ratio
                cost {value, currency}
                usage {value, unit}
              }
```

```
            }
          }
        }
      }
    }
  }
}
```

Response

The name of the category, the ratio of total usage attributed to that category, and the total cost and energy use attributed to that category are included in the response. The query also returns the ID of the billing account and service agreement.


```
{
  "data": {
    "billingAccountsConnection": {
      "edges": [
        {
          "node": {
            "utilityId": "123456",
            "serviceAgreementsConnection": {
              "edges": [
                {
                  "node": {
                    "utilityId": "1614570777",
                    "serviceType": "ELECTRICITY",
                    "consumptionCategories": [
                      {
                        "name": "APPLIANCES",
                        "ratio": 0.16,
                        "cost": {
                          "value": 87.61,
                          "currency": "USD"
                        },
                        "usage": {
                          "value": 386,
                          "unit": "KWH"
                        }
                      },
                      {
                        "name": "COOLING",
                        "ratio": 0.13,
                        "cost": {
                          "value": 72.28,
                          "currency": "USD"
                        },
                        "usage": {
                          "value": 318,
                          "unit": "KWH"
                        }
                      },
                      {
                        "name": "ELECTRONICS",
                        "ratio": 0.36,
```

```
      "cost": {
        "value": 198.99,
        "currency": "USD"
      },
      "usage": {
        "value": 876,
        "unit": "KWH"
      }
    },
    {
      "name": "LIGHTING",
      "ratio": 0.1,
      "cost": {
        "value": 56.51,
        "currency": "USD"
      },
      "usage": {
        "value": 249,
        "unit": "KWH"
      }
    },
    {
      "name": "OTHER",
      "ratio": 0.25,
      "cost": {
        "value": 132.18,
        "currency": "USD"
      },
      "usage": {
        "value": 581,
        "unit": "KWH"
      }
    },
    {
      "name": "REFRIGERATOR",
      "ratio": 0.16,
      "cost": {
        "value": 87.61,
        "currency": "USD"
      },
      "usage": {
        "value": 386,
        "unit": "KWH"
      }
    },
    {
      "name": "TOTAL",
      "ratio": 1,
      "cost": {
        "value": 547.57,
        "currency": "USD"
      },
      "usage": {
        "value": 2410,
        "unit": "KWH"
      }
```

```
                    }
                  ]
                }
              },
              {
                "node": {
                  "utilityId": "5441650371",
                  "serviceType": "GAS",
                  "consumptionCategories": [
                    {
                      "name": "TOTAL",
                      "ratio": 1,
                      "cost": {
                        "value": 210.99,
                        "currency": "USD"
                      },
                      "usage": {
                        "value": 19,
                        "unit": "TH"
                      }
                    },
                    {
                      "name": "WATER_HEATING",
                      "ratio": 1,
                      "cost": {
                        "value": 210.99,
                        "currency": "USD"
                      },
                      "usage": {
                        "value": 19,
                        "unit": "TH"
                      }
                    }
                  ]
                }
              }
            ]
          }
        }
      ]
    }
  }
}
```


Bill Forecast Example

Bill forecast information can be retrieved as shown in the following example.

Upcoming Bill Forecast Query Using Account-Scoped Access Tokens

The following query retrieves the upcoming bill forecast, including forecasts for the segments of the upcoming bill.

```
query GetBillForecast {
  billingAccountsConnection(first:10) {
    edges {
      node {
        utilityId
        uuid
        customerClass
        billForecast {
          timeInterval
          currentDateTime
          estimatedUsage {value, unit}
          estimatedUsageCharges {value, currency}
          estimatedUsageChargesUpperBound {value, currency}
          soFarUsage {value, unit}
          soFarUsageCharges {value, currency}
          priorYearUsage {value, unit}
          priorYearUsageCharges {value, currency}
          segments {
            serviceAgreement {
              serviceType
            }
            estimatedUsage {value, unit}
            estimatedUsageCharges {value, currency}
            estimatedUsageChargesUpperBound {value, currency}
            soFarUsage {value, unit}
            soFarUsageCharges {value, currency}
            priorYearUsage {value, unit}
            priorYearUsageCharges {value, currency}
          }
        }
      }
    }
  }
}
```

Response

The response includes energy use and costs that have been incurred so far, and also estimated energy use and costs based on projected energy use through the remaining portion of the bill period. Historical energy use and costs for the bill that occurred during the same time period in the previous year is also included.

```
{
  "data": {
    "billingAccountsConnection": {
      "edges": [
        {
          "node": {
            "utilityId": "123456",
```

```
            "uuid": "e62b20f4-2b9e-11e6-9abd-d2a42a76b0d6",
            "customerClass": "RESIDENTIAL",
            "billForecast": {
              "timeInterval":
"2024-11-05T00:00:00-05:00/2024-12-03T00:00:00-05:00",
              "currentDateTime": "2024-11-23T00:00:00-05:00",
              "estimatedUsage": {
                "value": 225,
                "unit": "EI"
              },
              "estimatedUsageCharges": {
                "value": 33,
                "currency": "USD"
              },
              "estimatedUsageChargesUpperBound": null,
              "soFarUsage": {
                "value": 122,
                "unit": "EI"
              },
              "soFarUsageCharges": {
                "value": 18,
                "currency": "USD"
              },
              "priorYearUsage": {
                "value": 129,
                "unit": "EI"
              },
              "priorYearUsageCharges": {
                "value": 43,
                "currency": "USD"
              },
              "segments": [
                {
                  "serviceAgreement": {
                    "serviceType": "ELECTRICITY"
                  },
                  "estimatedUsage": {
                    "value": 157,
                    "unit": "KWH"
                  },
                  "estimatedUsageCharges": {
                    "value": 25,
                    "currency": "USD"
                  },
                  "estimatedUsageChargesUpperBound": null,
                  "soFarUsage": {
                    "value": 98,
                    "unit": "KWH"
                  },
                  "soFarUsageCharges": {
                    "value": 15,
                    "currency": "USD"
                  },
                  "priorYearUsage": {
                    "value": 72,
                    "unit": "KWH"
```

```
            },
            "priorYearUsageCharges": {
              "value": 20,
              "currency": "USD"
            }
          },
          {
            "serviceAgreement": {
              "serviceType": "GAS"
            },
            "estimatedUsage": {
              "value": 8,
              "unit": "TH"
            },
            "estimatedUsageCharges": {
              "value": 8,
              "currency": "USD"
            },
            "estimatedUsageChargesUpperBound": null,
            "soFarUsage": {
              "value": 3,
              "unit": "TH"
            },
            "soFarUsageCharges": {
              "value": 3,
              "currency": "USD"
            },
            "priorYearUsage": {
              "value": 6,
              "unit": "TH"
            },
            "priorYearUsageCharges": {
              "value": 23,
              "currency": "USD"
            }
          }
        ]
      }
    }
  ]
}
}
```

6
Reference

Referential information on some of the API standards are available below.

Status Codes

The GraphQL API returns a standard HTTP status code in the Response header.

HTTP Status Code	Description
200 OK	The request was successfully completed.
202 Accepted	The request has been accepted for processing, but the processing has not been completed. The request may or may not eventually be acted upon, as it may be disallowed at the time processing takes place.
400 Bad Request	The request could not be processed because it contains missing or invalid information (such as, a validation error on an input field, a missing required value, and so on).
401 Unauthorized	The request is not authorized. The authentication credentials included with this request are missing or invalid.
403 Forbidden	The user cannot be authenticated. The user does not have authorization to perform this request.
404 Not Found	The request includes a resource URI that does not exist.
415 Not Acceptable	The client's ContentType header is not correct. For example, the client attempts to send the request in XML, but the resource can only accept JSON.
500 Internal Server Error	The server encountered an unexpected condition that prevented it from fulfilling the request.
503 Service Unavailable	The server is unable to handle the request due to temporary overloading or maintenance of the server. The web application is not currently running.

JSON Format

All responses are in JSON, and follow the guidelines of well-formed JSON.

Pagination

The Oracle Utilities GraphQL APIs use the standard connections pattern for pagination.

GraphQL Comparison to REST API

The following topics can assist users who are more familiar with REST APIs to be aware of how certain concepts and techniques work for GraphQL APIs:

- **Data Requests**:
  - REST API: The client makes a request to a specific endpoint and receives all the data associated with that endpoint
  - GraphQL API: The client sends a query specifying the data required, and only that data is returned.
- **Response Format**:
  - REST API: The response format is fixed and determined by the server.
  - GraphQL API: The client specifies the desired format of the response, allowing for more flexibility in the data returned.
- **Multiple Requests**:
  - REST API: It may be necessary to make multiple requests to different endpoints to get all the data needed.
  - GraphQL API: All the data can be retrieved with a single request, making it more efficient for certain use cases.
- **Versioning**:
  - REST API: Different versions of the API are typically created to maintain backward compatibility.
  - GraphQL API: New features can be added without affecting existing queries.
- **Caching**: REST APIs are built to leverage the network-level caching built into the internet, but GraphQL requests cannot be cached by intermediate network infrastructure. Since our GraphQL APIs work with account-specific data that usually changes with each request and would not be shared by many web sessions, the lack of caching support is not a significant downside for our APIs.

7

Contact Your Delivery Team

Your Oracle Delivery Team is the group responsible for setting up, configuring, launching, or expanding your Oracle Utilities Opower program. Contact your Delivery Team if you have any questions about your program products and implementation.

To contact your Delivery Team:

1. Sign in to Inside Opower (https://inside.opower.com). This is your portal for questions and information related to your program.
2. Go to the Community tab to see who is on your Delivery Team.
3. Contact any of the team members using the information provided.

If you need to report an issue or get technical support, contact My Oracle Support.