

ENGINES API - VERSION 3.0  
C# Migration Guide  
November 2023



## Contents

1.	Why v3? .....	3
2.	Summary of Changes .....	3
3.	Authentication .....	4
4.	Cache-Control .....	5
4.1	What is Cache-Control? .....	5
4.2	Using the Cache-Control Header .....	5
4.3	Default API Behavior .....	5
4.4	Setting Cache Control Via the .NET SDK .....	6
4.5	Standalone Request Example .....	6
4.6	Sample Use Cases .....	7
5.	Changes to Calculation Parameters Body .....	8
5.1	“data” Top-Level Object .....	8
5.2	“meta” Top-Level Object (Optional) .....	9
5.3	STACH v2 .....	9
5.4	SDK Changes .....	10
5.4.1	CalculationParametersRoot object .....	10
5.4.2	STACH Extensions SDK .....	10
6.	New and Modified Endpoints .....	11
6.1	New Endpoints .....	11
6.1.1	PUT and Calculate .....	11
6.1.2	GET Status By ID .....	12
6.1.3	GET All Calculations .....	13
6.1.4	PA Pricing Source Lookups .....	13
6.1.5	GET Quant Engine Calculation Metadata Information by Id .....	13
6.1.6	Fixed Income Calculations Discount Curve Lookups .....	13
6.2	Modified Endpoints .....	14
6.2.1	POST Calculation .....	14
6.2.2	GET Calculation Parameters By ID .....	14
6.2.3	GET Calculation Unit Results By ID .....	15
6.2.4	DELETE Calculation By ID .....	15
6.2.5	Components Lookup .....	16
6.2.6	Documents Lookup .....	16
6.2.7	Frequency Lookup .....	17
6.2.8	Dates Lookup .....	17
6.2.9	PA Columns Lookup .....	17
6.2.10	PA Groups Lookup .....	18
6.2.11	PA Column Statistics Lookup .....	18
6.2.12	Vault Configurations Lookup .....	18
6.2.13	SPAR Benchmarks Lookup .....	18
6.2.14	Accounts Lookup .....	18
6.2.15	Currencies Lookup .....	19
7.	Interactive Endpoints .....	19
7.1	Setting X-FactSet-Api-Long-Running-Deadline header Via the .NET SDK .....	19
7.2	Standalone Request Example .....	20
8.	Installing v3-Compatible .NET SDK .....	21
9.	Troubleshooting .....	22
10.	Appendix .....	22

## 1. Why v3?

The Engines API houses many of the calculation-based APIs in the Analytics API program, including PA, SPAR, Vault, Publisher, Fixed Income, and Optimizer. Version 2 was released in August 2019 and combined the previously distinct 'Batch API' with the single-request v1 Engine APIs.

Engines v3 improves both the developer experience and engine calculation efficiency with a light rework of engines endpoints, focusing on increased user control of the calculation flow. The Analytics API program designed v3 with the following overarching goals in mind:

1. Create more 'RESTful' API endpoints by treating submitted calculation parameters as a user resource that allow for CRUD operations (Create, Read, Update, Delete).
2. Improve calculation workflow efficiency (i.e., fewer redundant calculations).
3. Align with various FactSet-wide design standards.
4. Support for OAuth2.0 which can be setup through FactSet's Developer Portal.

## 2. Summary of Changes

You can expect the following changes in v3 of the Engines APIs:

1. FactSet Enterprise SDK supports two forms of authentication -
  - a. Basic Auth or API Key
  - b. OAuth2.0
2. Support for cache-control header to give clients the ability to fetch pre-calculated results for a specified period (up to 12 hours), or explicitly request an ad hoc calculation.
  - a. By default, all requests will be stored for 12 hours.
3. Updates to the calculation parameters (POST and PUT body) to give clients the option to choose between the different response formats offered with STACH v2 (for more information on STACH v2, refer to the documentation [here](#) and [here](#)).
4. Updated endpoint URLs to be engine-specific.
  - a. v3: POST `analytics/engines/{engine1}/v3/calculations`
  - b. v2: POST `analytics/engines/v2/calculations`
5. Support for user-provided calculation ids for polling and result URL construction via a new PUT method.
  - a. E.g. You can now create a calculation request and specify a custom "daily\_report" calculation id via the new PUT endpoint. This will allow you to directly construct the corresponding status polling and result pickup URLs:
    - i. [https://api.factset.com/analytics/engines/{engine}/v3/calculations/daily\\_report/status](https://api.factset.com/analytics/engines/{engine}/v3/calculations/daily_report/status)
    - ii. [https://api.factset.com/analytics/engines/{engine}/v3/calculations/daily\\_report/units/{unitId}/result](https://api.factset.com/analytics/engines/{engine}/v3/calculations/daily_report/units/{unitId}/result)
6. Support for interactive endpoints that return calculation results as a response to the initial POST/PUT request if the calculation has 1 unit and it completes within a short period of time, defined per request.

---

<sup>1</sup> Where engine can be pa, spar, pub, vault, or fi.

### 3. Authentication

The FactSet Enterprise SDK supports two forms of authentication, both must be setup through [FactSet's Developer Portal](#).

1. [OAuth2.0](#) – (preferred).
2. [API Key](#)

```

1  using System;
2  using System.Net;
3  using System.Threading.Tasks;
4  using FactSet.SDK.Utils.Authentication;
5  using FactSet.SDK.PAEngine.Api;
6  using FactSet.SDK.PAEngine.Client;
7  using FactSet.SDK.PAEngine.Model;
8
9  namespace Example
10 {
11     public class PostAndCalculateExample
12     {
13         public static async Task Main()
14         {
15             var config = new FactSet.SDK.PAEngine.Client.Configuration();
16
17             // Examples for each supported authentication method are below,
18             // choose one that satisfies your use case.
19
20             /* (Preferred) OAuth 2.0: FactSetOAuth2 */
21             // See https://github.com/FactSet/enterprise-sdk#oauth-20
22             // for information on how to create the app-config.json file
23             // See https://github.com/FactSet/enterprise-sdk-utils-dotnet#authentication
24             // for more information on using the ConfidentialClient class
25             ConfidentialClient confidentialClient = await ConfidentialClient.CreateAsync("/path/to/app-config.json");
26             config.OAuth2Client = confidentialClient;
27
28             /* Basic authentication: FactSetApiKey */
29             // See https://github.com/FactSet/enterprise-sdk#api-key
30             // for information how to create an API key
31             // config.Username = "USERNAME-SERIAL";
32             // config.Password = "API-KEY";
33
34             var apiInstance = new PACalculationsApi(config);
35
36             var xFactSetApiLongRunningDeadline = 10; // int? | Long running deadline in seconds when only one unit is passed in the POST body. (optional)
37             var cacheControl = "cacheControl_example"; // string | Standard HTTP header. Accepts max-stale. (optional)
38             var pACalculationParametersRoot = new PACalculationParametersRoot(); // PACalculationParametersRoot | Calculation Parameters (optional)
39
40             try
41             {
42                 // Create and Run PA calculation
43                 PACalculationsApi.PostAndCalculateResponseWrapper result = apiInstance.PostAndCalculate(xFactSetApiLongRunningDeadline,
44                 cacheControl, pACalculationParametersRoot);
45
46                 switch (result.StatusCode)
47                 {

```

## 4. Cache-Control

v3 includes support for the Cache-Control header in the [POST](#) and [PUT](#) calculation endpoints by accepting a max-stale value.

### 4.1 What is Cache-Control?

The Cache-Control HTTP header holds directives (instructions) for caching both requests and responses. Engines APIs expose access to the max-stale directive to allow for calculation request caching. For more information on Cache-Control, refer to the documentation [here](#).

### 4.2 Using the Cache-Control Header

Setting “max-stale=<staleness limit in seconds>” allows clients to fetch pre-calculated results with any subsequent POST requests<sup>2</sup> if they were last calculated within the staleness limit. The max cache-control value is “max-stale=43200” (12 hours).

Once set, the API will check to see if the stored results are within the staleness limit.

- If they are within the limit AND the request contains only 1 calculation unit:
  - a. Then, the request will return a “201” response code with the results in the body of the POST/PUT response AND the status polling URL in the Location header. This allows for a quick response and eliminates unnecessary points accrual.
- If they are within the limit AND the request contains multiple calculation units:
  - a. Then, the request will return a “200” response code with the result URLs in the body of the POST/PUT response. This allows for a quick response and eliminates unnecessary points accrual.
- If the results are not within the staleness limit, a brand-new calculation request will be triggered to get the latest results. Note that the caching period starts after the individual calculation unit has completed.

To immediately request the latest results, override the cache by setting “max-stale=0” in the Cache-control header parameter. ***Changes made to the underlying PA or SPAR document via the workstation will not trigger result re-calculation. The cache-control parameter should be set to “max-stale=0 to immediately request the latest results.***

Note that the max-stale value only controls when a new calculation will be triggered with any subsequent POST or PUT requests (provided that the request parameters stay the same). All results generated will always be stored for 12 hours, the max-stale value will not affect this.

### 4.3 Default API Behavior

*By default, all results are cached for 12 hours.* This means that without sending any cache-control headers, after the first request successfully completes, all subsequent requests with unchanged request parameters will return the same results for 12 hours.

---

<sup>2</sup> The subsequent POST requests must have the same request body. Any changes in the request body will trigger a new calculation.

#### 4.4 Setting Cache Control Via the .NET SDK

- Create a String cacheControl parameter and set to “max-stale=0”– line 37 below
- Pass cacheControl parameter in the PostAndCalculate() or PutAndCalculate() methods – line 44 below
  - Note that all engines calculations APIs (PACalculationsApi, SPARCalculationsApi, VaultCalculationsApi, QuantCalculationsApi, FICalculationsApi) support the cacheControl parameter via the PostAndCalculate() and PutAndCalculate() endpoints.

```

13 public static async Task Main()
14 {
15     var config = new FactSet.SDK.PAEngine.Client.Configuration();
16
17     // Examples for each supported authentication method are below,
18     // choose one that satisfies your use case.
19
20     /* (Preferred) OAuth 2.0: FactSetOAuth2 */
21     // See https://github.com/FactSet/enterprise-sdk#oauth-20
22     // for information on how to create the app-config.json file
23     // See https://github.com/FactSet/enterprise-sdk-utils-dotnet#authentication
24     // for more information on using the ConfidentialClient class
25     ConfidentialClient confidentialClient = await ConfidentialClient.CreateAsync("/path/to/app-config.json");
26     config.OAuth2Client = confidentialClient;
27
28     /* Basic authentication: FactSetApiKey */
29     // See https://github.com/FactSet/enterprise-sdk#api-key
30     // for information how to create an API key
31     // config.Username = "USERNAME-SERIAL";
32     // config.Password = "API-KEY";
33
34     var apiInstance = new PACalculationsApi(config);
35
36     var xFactSetApiLongRunningDeadline = 10; // int? | Long running deadline in seconds when only one unit is passed in the POST body. (optional)
37     var cacheControl = "cacheControl_example"; // string | Standard HTTP header. Accepts max-stale. (optional)
38     var pACalculationParametersRoot = new PACalculationParametersRoot(); // PACalculationParametersRoot | Calculation Parameters (optional)
39
40     try
41     {
42         // Create and Run PA calculation
43         PACalculationsApi.PostAndCalculateResponseWrapper result = apiInstance.PostAndCalculate(xFactSetApiLongRunningDeadline,
44             cacheControl, pACalculationParametersRoot);
45
46         switch (result.StatusCode)
47         {
48             { ...
49         }
50     }
51     catch (ApiException e)
52     {
53         Console.WriteLine("Exception when calling PACalculationsApi.PostAndCalculate: " + e.Message);
54         Console.WriteLine("Status Code: " + e.ErrorCode);
55         Console.WriteLine(e.StackTrace);
56     }
57 }
58 }
59 }
60 }
61 }
62 }
63 }
64 }
65 }
66 }
67 }
68 }
69 }
70 }
71 }

```

#### 4.5 Standalone Request Example

**Standalone Request:**  
 POST https://api.factset.com/analytics/engines/pa/v3/calculations

**Request Headers:**  
 Content-Type:  
 application/json  
 Cache-Control: max-stale=0  
 Authorization:Basic\*\*\*\*\*

```

Accept: */*
Host: api.factset.com
Accept-Encoding: gzip, deflate,br
Content-Length: 739

Body:
{
  "data": {
    "1": {
      "componentid": "125872E5D836683A3EB09EF8C73D3E1BE854BE4EF6173BAB4BFBB5752788F4E9",
      "accounts": [
        {
          "id": "BENCH:SP50",
          "holdingsmode": "B&H",
        }
      ],
      "dates": {
        "startdate": "20170101",
        "enddate": "20180101",
        "frequency": "Monthly",
      },
      "groups": [
        {
          "id": "E5DB59F608778844CD784DD5659F65AA8A0A9F63F9E4314B2D92761AF0B1B3D9"
        }
      ],
      "benchmarks": [
        {
          "id": "BENCH:R.1000"
        }
      ],
      "currencyisocode": "USD"
    }
  }
}

```

## 4.6 Sample Use Cases

### Portfolio Manager looks at daily report on internal portal at 1pm

- POST calculation parameters for a multi-unit calculation to `analytics/engines/spar/v3/calculations` and set `"max-stale=43200"` (12-hour request cache)
  - A new calculation request is initiated, and the calculation successfully completes.
  - User receives a 202, polls the status endpoint, and fetches results from the results endpoint.
- That afternoon, user sends another POST to `"/analytics/engines/spar/v3/calculations"` with the same request body.
  - A new calculation request is not initiated because the report was run within the past 12 hours.
  - User receives a 200 and fetches results from the result URLs returned in the POST response body.
- The next day, user sends another POST to `"/analytics/engines/spar/v3/calculations"` with the same request body.

- o A new calculation request is initiated because the report was initially run more than 12 hours ago.
- o User receives a 202, polls the status endpoint, and fetches results from the results endpoint.

**3 hours later, a new column is added to the report via the SPAR UI**

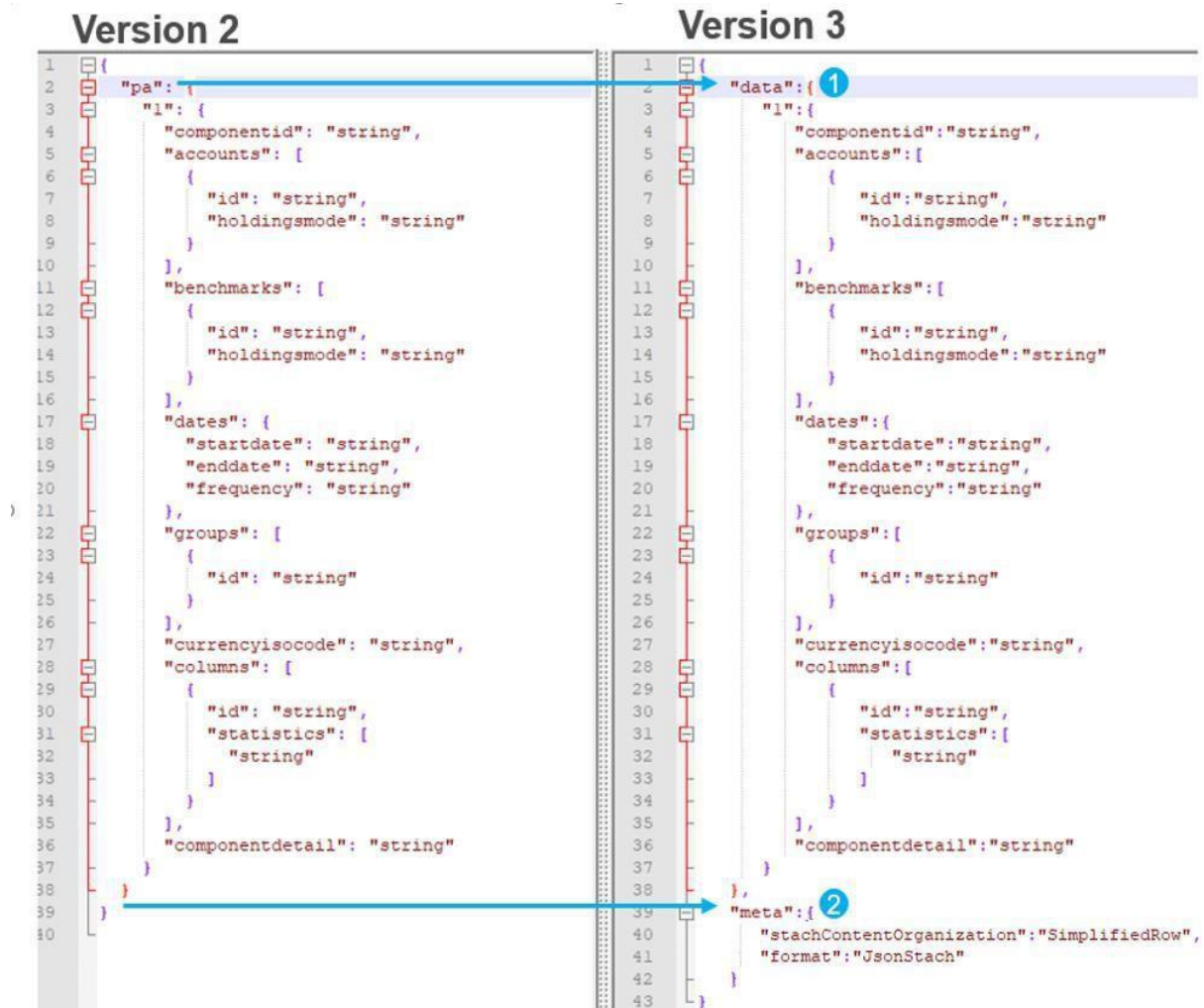
- POST calculation parameters to “analytics/engines/spar/v3/calculations” calculations and set “max-stale=0” to receive the latest results.
  - o A new calculation request is initiated because the report has been updated.
  - o User receives a 202, polls the status endpoint, and fetches results from the results endpoint.

## 5. Changes to Calculation Parameters Body

### 5.1 “data” Top-Level Object

The existing POST body and the new PUT request body will move under a new top-level object, “data”, replacing the engine-specific object.

In the example below, you can see the “pa” object in line 2 has been replaced with “data”.





## 5.2 “meta” Top-Level Object (Optional)

A new optional “meta” object will be available for clients to specify the response’s STACH v2 format. The “meta” object includes a “**stachContentorganization**” parameter to specify the STACH v2 format to return in the response, and a “**format**” parameter that allows user to specify whether the response should be in Json or Binary.

## 5.3 STACH v2

v2 of STACH introduced support for the new row organized format and simplified row format, along with the column organized format supported in the previous version of STACH. For more information on the new STACH v2 format in general, refer to the official documentation [here](#). For Engines-specific documentation and samples, refer to the documentation [here](#).

Note that STACH v2 will be the only version of STACH supported by the v3 Engines APIs, however STACH v2 supports the previous STACH v1 column organized format via the “stachContentorganization” parameter in the “meta” object.

Note that if the “meta” object is not specified, the default stachContentOrganization is SimplifiedRow and the default format is Json (STACH).

There is no other change in the calculation parameters from v2 to v3. The screenshot below is an excerpt from the PA Engine API Swagger hosted on [Developer Portal](#). It shows the PA calculation parameter schema that describes the PA request body required and optional fields.

```

PACalculationParametersRoot v {
  data > {...}
  meta CalculationMeta v {
    stachContentOrganization string
                                     default: SimplifiedRow
                                     Enum:
                                     > Array [ 4 ]
    format string
                                     default: Json
                                     Enum:
                                     > Array [ 2 ]
  }
}

```

## 5.4 SDK Changes

### 5.4.1 CalculationParametersRoot object

In previous versions of the SDK, calculation parameters were constructed by creating an <Engine>CalculationParameters object.

In the latest version of the Engines SDK for v3, the <Engine>CalculationParameters object is replaced by the <Engine>**CalculationParametersRoot** object (line 38 in the screenshot below) which contains the new “**data**” and “**meta**” objects and all the relevant models.

```

33
34     var apiInstance = new PACalculationsApi(config);
35
36     var xFactSetApiLongRunningDeadline = 10; // int? | Long running deadline in seconds when only one unit is passed in the POST body. (optional)
37     var cacheControl = "cacheControl_example"; // string | Standard HTTP header. Accepts max-stale. (optional)
38     var pACalculationParametersRoot = new PACalculationParametersRoot(); // PACalculationParametersRoot | Calculation Parameters (optional)
39
40     try
41     {
42         // Create and Run PA calculation
43         PACalculationsApi.PostAndCalculateResponseWrapper result = apiInstance.PostAndCalculate(xFactSetApiLongRunningDeadline,
44             cacheControl, pACalculationParametersRoot);
45
46         switch (result.StatusCode)
47         {
48             case (HttpStatusCode)200:
49                 Console.WriteLine(result.Response200);
50                 break;
51             case (HttpStatusCode)201:
52                 Console.WriteLine(result.Response201);
53                 break;
54             case (HttpStatusCode)202:
55                 Console.WriteLine(result.Response202);
56                 break;
57         }
58     }
59     catch (ApiException e)
60     {
61         Console.WriteLine("Exception when calling PACalculationsApi.PostAndCalculate: " + e.Message );
62         Console.WriteLine("Status Code: "+ e.ErrorCode);
63         Console.WriteLine(e.StackTrace);
64     }
65 }
66 }
67 }
68 }
69 }
70 }
71 }

```

### 5.4.2 STACH Extensions SDK

The [STACH Extensions Package](#) has been updated to include support for converting STACH v2 into a dataframe. The FactSet Enterprise .Net SDK v1.0.0 for API Engines does not include the latest STACH Extensions package, hence the user need to install the .NET stach extensions package from [here](#).

The code snippet below shows how to use the STACH extensions library to convert both STACH v2 and v1 to Table format.

```

--
25 // Stach v2 Row Organized format
26 var rowStachBuilder = StachExtensionFactory.GetRowOrganizedBuilder();
27 rowStachBuilder.SetPackage(data); // data is the stach input in string or object format
28 // var rowOrganizedPackage = rowStachBuilder.GetPackage();
29 var stachExtension = rowStachBuilder.Build();
30 var table = stachExtension.ConvertToTable();
31
32
33 // Stach v2 Column Organized format
34 var columnStachBuilder = StachExtensionFactory.GetColumnOrganizedBuilder<Stach.V2.Package>();
35 columnStachBuilder.SetPackage(data); // data is the stach input in string or object format
36 // var package = columnStachBuilder.GetPackage();
37 var stachExtension = columnStachBuilder.Build();
38 var table = stachExtension.ConvertToTable();
39
40
41 // Stach v1 Column Organized format
42 var columnStachBuilder = StachExtensionFactory.GetColumnOrganizedBuilder<Package>();
43 columnStachBuilder.SetPackage(data); // data is the stach input in string or object format
44 // var package = columnStachBuilder.GetPackage();
45 var stachExtension = columnStachBuilder.Build();
46 var table = stachExtension.ConvertToTable();
47

```

<https://github.com/factset/stach-extensions#dotnet-1>

## 6. New and Modified Endpoints

### 6.1 New Endpoints

#### 6.1.1 PUT and Calculate

PUT analytics/engines/{engine}/v3/calculations/{id}

- New method that allows clients to create and update custom calculation ids. It creates/updates and runs the engine calculation specified in the request body.
- `Cache-control` header functionality as outlined [above](#).
- “data” and “meta” top level objects as outlined [above](#).
- Interactive endpoint functionality as outlined [here](#).
- SDK method and models:
  - New PutAndCalculate() method that takes in a String id (required), an <Engine>CalculationParametersRoot object (required), a cacheControl String (optional), and a xFactSetApiLongRunningDeadline Integer (optional)
  - Response Object models:
    - 202 Response: [CalculationStatusRoot](#)
    - 200 Response: [CalculationStatusRoot](#)
    - 201 Response: [ObjectRoot](#)
  - You can find engine-specific documentation in the links below:

API-Specific Method	Request Object Model
<a href="#">PACalculationsApi.PutAndCalculate()</a>	<a href="#">PACalculationParametersRoot</a>
<a href="#">SPARCalculationsApi.PutAndCalculate()</a>	<a href="#">SPARCalculationParametersRoot</a>

<a href="#">VaultCalculationsApi.PutAndCalculate()</a>	<a href="#">VaultCalculationParametersRoot</a>
<a href="#">QuantCalculationsApi.PutAndCalculate()</a>	<a href="#">QuantCalculationParametersRoot</a>
<a href="#">FICalculationsApi.PutAndCalculate()</a>	<a href="#">FICalculationParametersRoot</a>

**Sample Use cases:**

**Portfolio Manager looks at a daily report on an internal portal at 9am**

- PUT calculation parameters for a multi-unit calculation to **analytics/engines/pa/v3/calculations/daily\_report** and set “max-stale=43200” (12-hour request cache)
  - A new calculation request is initiated, and the calculation successfully completes. The calculation parameters are successfully saved to the ‘daily\_report’ calculation id.
  - User receives a 202
  - User polls the pre-constructed status URL:
    - [https://api.factset.com/analytics/engines/pa/v3/calculations/daily\\_report/status](https://api.factset.com/analytics/engines/pa/v3/calculations/daily_report/status)
  - User fetches results from the pre-constructed results URL:
    - [https://api.factset.com/analytics/engines/pa/v3/calculations/daily\\_report/units/{unitId}/result](https://api.factset.com/analytics/engines/pa/v3/calculations/daily_report/units/{unitId}/result)

**Two hours later, the report is updated to include a new benchmark.**

- PUT calculation parameters to **analytics/engines/pa/v3/calculations/daily\_report** and set “max-stale=43200” (12-hour request cache)
  - A new calculation request is initiated since the request parameters have changed, and the calculation successfully completes. The calculation parameters are successfully updated to the ‘daily\_report’ calculation id.
  - User receives a 202, polls the pre-constructed status URL, and fetches results from the pre-constructed results URL.
  - Results include the newly-added benchmark data.
- That afternoon, user sends another PUT to **analytics/engines/pa/v3/calculations/daily\_report** with the same request body.
  - A new calculation request is not initiated because the report was run within the past 12 hours and the request parameters are unchanged.
  - User receives a 200 and fetches stored results from the pre-constructed result URLs returned in the PUT response body.
- The next day, user sends another PUT to **analytics/engines/pa/v3/calculations/daily\_report** with the same request body.
  - A new calculation request is initiated because the report was last run more than 12 hours ago.
  - User receives a 202, polls the pre-constructed status URL, and fetches results from the pre-constructed results URL.

**6.1.2 GET Status By ID**

GET `analytics/engines/{engine}/v3/calculations/{id}/status`

- New endpoint to get calculation status, will function the same as GET `{engine}/calculations/{id}` did in v2.
- If the calculation has finished computing, the location header will point to the result URL. Otherwise, if the calculation is still running and the X-FactSet-Api-PickUp-Progress header will contain a progress percentage.
- Status field in the response points to the state of the calculation. It can take three values, “Queued”, “In Progress” and “Completed”.
- Response will include a parent object “data.”

- SDK method and models:
  - New GetCalculationStatusById(id) method that takes in a String id (required)
  - [CalculationStatusRoot](#) response object model for all response codes
  - You can find engine-specific documentation in the links below:

<b>API-Specific Method</b>
<a href="#">PACalculationsApi.GetCalculationStatusById(id)</a>
<a href="#">SPARCalculationsApi.GetCalculationStatusById(id)</a>
<a href="#">VaultCalculationsApi.GetCalculationStatusById(id)</a>
<a href="#">QuantCalculationsApi.GetCalculationStatusById(id)</a>
<a href="#">FICalculationsApi.GetCalculationStatusById(id)</a>

### 6.1.3 GET All Calculations

GET `analytics/engines/{engine}/v3/calculations`

- New endpoint which returns all the calculations.
- V3: analytics/engines/pa/v3/calculations
- This endpoint is available only for PA, SPAR, Vault, and Quant Engine APIs.

<b>API-Specific Method</b>
<a href="#">PACalculationsApi.GetAllCalculations(pageNumber)</a>
<a href="#">SPARCalculationsApi.GetAllCalculations(pageNumber)</a>
<a href="#">VaultCalculationsApi.Get.GetAllCalculations(pageNumber)</a>
<a href="#">QuantCalculationsApi.GetAllCalculations(pageNumber)</a>

### 6.1.4 PA Pricing Source Lookups

GET `/analytics/engines/pa/v3/pricing-sources`

- This endpoint lists all the PA pricing sources that can be applied to a PA calculation.
- V3: analytics/engines/pa/v3/pricing-sources
- API Specific Method - [PricingSourcesApi.GetPAPricingSources\(\)](#)
- New response Object model - [PAPricingSourceRoot](#)

### 6.1.5 GET Quant Engine Calculation Metadata Information by Id

GET `/analytics/engines/quant/v3/calculations/{id}/units/{unitId}/info`

- This endpoint is used to get the metadata information of a previously requested calculation.
- V3: analytics/engines/quant/v3/calculations/{id}/units/{unitId}/info
- API Specific Method - [QuantCalculationsApi.GetCalculationUnitInfoById](#)

### 6.1.6 Fixed Income Calculations Discount Curve Lookups

GET `/analytics/engines/fi/v3/discount-curves`

- This endpoint lists all the discount curves that can be applied to a FI calculation.

- V3: analytics/engines/fi/v3/discount-curves
- API Specific Method - [DiscountCurvesApi.GetAllFIDiscountCurves](#)
- New response Object model - [FIDiscountCurveInfoRoot](#)

## 6.2 Modified Endpoints

### 6.2.1 POST Calculation

POST `analytics/engines/{engine}/v3/calculations`

- `Cache-control` header functionality as outlined [above](#).
- “data” and “meta” top level objects as outlined [above](#).
- New response object models
- Interactive endpoint functionality as outlined [here](#).
- SDK changes:
  - New `PostAndCalculate()` method that takes in an `<Engine>CalculationParametersRoot` object (required), a `cacheControl` String (optional), and a `xFactSetApiLongRunningDeadline` Integer (optional)
    - Replaces the `runPACalculation()` method in the previous major version of the SDK
  - Response Object models:
    - 202 Response: [CalculationStatusRoot](#)
    - 200 Response: [CalculationStatusRoot](#)
    - 201 Response: [ObjectRoot](#)
  - You can find engine-specific documentation in the links below:

API-Specific Method	Request Object Model
<a href="#">PACalculationsApi.PostAndCalculate()</a>	<a href="#">PACalculationParametersRoot</a>
<a href="#">SPARCalculationsApi.PostAndCalculate()</a>	<a href="#">SPARCalculationParametersRoot</a>
<a href="#">VaultCalculationsApi.PostAndCalculate()</a>	<a href="#">VaultCalculationParametersRoot</a>
<a href="#">QuantCalculationsApi.PostAndCalculate()</a>	<a href="#">QuantCalculationParametersRoot</a>
<a href="#">FICalculationsApi.PostAndCalculate()</a>	<a href="#">FICalculationParametersRoot</a>

### 6.2.2 GET Calculation Parameters By ID

GET `analytics/engines/{engine}/v3/calculations/{id}`

- Now returns calculation parameters instead of calculation status.
- Takes in a calculation id string as a path parameter.
- The calculation id is user-defined when created via the PUT request.
- Calculation id is always returned in:
  - The location header of the POST/PUT response.
  - The body of a successful POST/PUT request.
- SDK changes:
  - New `GetCalculationParameters()` method that takes in a String id (required)

- New Response Object models: engine-specific CalculationParametersRoot object models
- You can find engine-specific documentation in the links below:

API-Specific Method	New Response Object Model
<a href="#">PACalculationsApi.GetCalculationParameters()</a>	<a href="#">PACalculationParametersRoot</a>
<a href="#">SparCalculationsApi.GetCalculationParameters()</a>	<a href="#">SPARCalculationParametersRoot</a>
<a href="#">VaultCalculationsApi.GetCalculationParameters()</a>	<a href="#">VaultCalculationParametersRoot</a>
<a href="#">QuantCalculationsApi.GetCalculationParameters()</a>	<a href="#">QuantCalculationParametersRoot</a>
<a href="#">FICalculationsApi.GetCalculationParameters()</a>	<a href="#">FICalculationParametersRoot</a>

### 6.2.3 GET Calculation Unit Results By ID

GET `analytics/engines/{engine}/v3/calculations/{id}/units/{unitId}/result`

- Functionally the same as v2, with result longevity dictated by PUT/POST cache-control headers.
- Added new SDK method to support result retrieval.
- Response will include a parent object “data.”
- New URL format:
  - V2: `/analytics/engines/v2/calculations/{id}/units/{unitid}/result`
  - V3: `/analytics/engines/pa/v3/calculations/{id}/units/{unitid}/result`
  - For Fixed Income v3 - `/analytics/engines/fi/v3/calculations/{id}/result`
- SDK changes:
  - New `GetCalculationUnitResultById()` method that takes in a String id (required), and a String unitId (required)
  - Response Object model: [ObjectRoot](#)
  - You can find engine-specific documentation in the links below:

API-Specific Method
<a href="#">PACalculationsApi.GetCalculationUnitResultById()</a>
<a href="#">SparCalculationsApi.GetCalculationUnitResultById()</a>
<a href="#">VaultCalculationsApi.GetCalculationUnitResultById()</a>
<a href="#">QuantCalculationsApi.GetCalculationUnitResultById()</a>
<a href="#">FICalculationsApi.GetCalculationResult()</a>

### 6.2.4 DELETE Calculation By ID

GET `analytics/engines/{engine}/v3/calculations/{id}`

- Functionally is same as v2, used to cancel a previously submitted calculation using the calculationId as input.
- A successful response will fetch a 204 status code.
- New URL format:
  - V2: `/analytics/engines/v2/calculations/{id}`
  - V3: `/analytics/engines/pa/v3/calculations/{id}`

API-Specific Method
<a href="#">PACalculationsApiCancelCalculationById(id)</a>
<a href="#">SPARCalculationsApiCancelCalculationById(id)</a>
<a href="#">VaultCalculationsApiCancelCalculationById(id)</a>
<a href="#">QuantCalculationsApiCancelCalculationById(id)</a>
<a href="#">FICalculationsApiCancelCalculationById(id)</a>

### 6.2.5 Components Lookup

GET `analytics/engines/{engine}/v3/components`

- V2: analytics/lookups/v2/engines/pa/components
- V3: analytics/engines/pa/v3/components
- SDK changes:
  - No change in the method names
  - Changes in the response object models:

Engine	API-Specific Method	New response object model
PA	<a href="#">ComponentsApi.GetPAComponents()</a>	<a href="#">ComponentSummaryRoot</a>
	<a href="#">ComponentsApi.GetPAComponentById()</a>	<a href="#">PAComponentRoot</a>
SPAR	<a href="#">ComponentsApi.GetSPARComponents()</a>	<a href="#">ComponentSummaryRoot</a>
	<a href="#">ComponentsApi.GetSPARComponentById()</a>	<a href="#">SPARComponentRoot</a>
Vault	<a href="#">ComponentsApi.GetVaultComponents()</a>	<a href="#">ComponentSummaryRoot</a>
	<a href="#">ComponentsApi.GetVaultComponentById()</a>	<a href="#">VaultComponentRoot</a>

### 6.2.6 Documents Lookup

GET `analytics/engines/{engine}/v3/document/{path}`

- V2: analytics/lookups/v2/engines/{engine}/documents/{path}
- V3: analytics/engines/{engine}/v3/documents/{path}
- SDK changes:
  - No change in the method name
  - New Response Object model: [DocumentDirectoriesRoot](#)

Engine	API-Specific Method	New response object model
PA	<a href="#">DocumentsApi.GetPA3Documents()</a>	<a href="#">DocumentDirectoriesRoot</a>
SPAR	<a href="#">DocumentsApi.GetSPAR3Documents()</a>	<a href="#">DocumentDirectoriesRoot</a>
Vault	<a href="#">DocumentsApi.GetVaultDocuments()</a>	<a href="#">DocumentDirectoriesRoot</a>



### 6.2.7 Frequency Lookup

GET `analytics/engines/{engine}/v3/frequencies`

- V2: analytics/lookups/v2/engines/{engine}/frequencies
- V3: analytics/engines/{engine}/v3/frequencies
- SDK changes:
  - No change in the method name
  - New Response Object model: [FrequencyRoot](#)

Engine	API-Specific Method	New response object model
PA	<a href="#">FrequenciesApi.GetPAFrequencies()</a>	<a href="#">FrequencyRoot</a>
SPAR	<a href="#">FrequenciesApi.GetSPARFrequencies()</a>	<a href="#">FrequencyRoot</a>
Vault	<a href="#">FrequenciesApi.GetVaultFrequencies()</a>	<a href="#">FrequencyRoot</a>

### 6.2.8 Dates Lookup

GET `analytics/engines/{engine}/v3/dates`

- V2: analytics/lookups/v2/engines/{engine}/dates
- V3: analytics/engines/{engine}/v3/dates
- SDK changes:
  - No change in the method name
  - New Response Object model: [DateParametersSummaryRoot](#)

Engine	API-Specific Method	New response object model
PA	<a href="#">DatesApi.ConvertPADatesToAbsoluteFormat(enddate, componentid, account)</a>	<a href="#">DateParametersSummaryRoot</a>
Vault	<a href="#">DatesApi.convertVaultDatesToAbsoluteFormat(enddate, componentid, account)</a>	<a href="#">DateParametersSummaryRoot</a>

### 6.2.9 PA Columns Lookup

GET `analytics/engines/pa/v3/columns/{id}`

- V2: analytics/lookups/v2/engines/pa/columns/
- V3: analytics/engines/pa/v3/columns
- SDK changes:
  - No change in the method name
  - New Response Object model:
    - [GetPAColumnById\(\): ColumnRoot](#)
    - [GetPAColumns\(\): ColumnSummaryRoot](#)

### 6.2.10 PA Groups Lookup

GET `analytics/engines/pa/v3/groups`

- V2: analytics/lookups/v2/engines/pa/groups
- V3: analytics/engines/pa/v3/groups
- SDK changes:
  - No change in the method name
  - New Response Object model:
    - [GetPAGroups\(\): GroupRoot](#)
    - [GetPAGroupingFrequencies\(\): FrequencyRoot](#)

### 6.2.11 PA Column Statistics Lookup

GET `analytics/engines/pa/v3/columnstatistics`

- V2: analytics/lookups/v2/engines/pa/columnstatistics
- V3: analytics/engines/pa/v3/columnstatistics
- SDK changes:
  - No change in the method name
  - New Response Object model:
    - [GetPAColumnStatistics\(\): ColumnStatisticRoot](#)

### 6.2.12 Vault Configurations Lookup

GET `/analytics/engines/vault/v3/configurations`

- V2: analytics/lookups/v2/engines/vault/configurations
- V3: analytics/engines/vault/v3/configurations
- SDK changes:
  - New Response Object model:
    - [GetVaultConfigurations\(account\): VaultConfigurationSummaryRoot](#)
    - [GetVaultConfigurationById\(id\): VaultConfigurationRoot](#)

### 6.2.13 SPAR Benchmarks Lookup

GET `/analytics/engines/spar/v3/benchmarks`

- V2: analytics/lookups/v2/engines/spar/benchmarks
- V3: analytics/engines/spar/v3/benchmarks
- SDK changes:
  - New Response Object model:
    - [GetSPARBenchmarkById\(id\): SPARBenchmarkRoot](#)

### 6.2.14 Accounts Lookup

GET `analytics/lookups/v3/accounts`

- V2: analytics/lookups/v2/accounts/
- V3: analytics/lookups/v3/accounts/
- SDK changes:
  - No change in the method name
  - New Response Object model: [AccountDirectoriesRoot](#)

Engine	API-Specific Method	New response object model
PA	<a href="#">AccountsApi.GetAccounts(path)</a>	<a href="#">AccountDirectoriesRoot</a>
SPAR	<a href="#">AccountsApi.GetAccounts(path)</a>	<a href="#">AccountDirectoriesRoot</a>
SPAR	<a href="#">AccountsApi.GetSPARReturnsType(accountPath)</a>	<a href="#">SPARAccountsRoot</a>

### 6.2.15 Currencies Lookup

GET `analytics/lookups/v3/currencies`

- V2: analytics/lookups/v2/engines/pa/currencies
- V3: analytics/lookups/v3/currencies
- New SDK method and models:
  - New GetCurrencies() method
    - Replaces the previous GetPACurrencies() method in the previous major version of the SDK
  - New response Object model: [CurrencyRoot](#)

Engine	API-Specific Method	New response object model
PA	<a href="#">CurrenciesApi.GetCurrencies()</a>	<a href="#">CurrencyRoot</a>
SPAR	<a href="#">CurrenciesApi.GetCurrencies()</a>	<a href="#">CurrencyRoot</a>

## 7. Interactive Endpoints

V3 Includes support for interactive POST and PUT endpoints. These will return calculation results directly as a response to the initial POST/PUT request if the calculation has 1 unit and it completes within a specified period defined per request (20 seconds or less). Results will still be available via the

GET `analytics/engines/{engine}/v3/calculations/{id}/units/{unitId}/result` endpoint.

- If the calculation completes successfully, a user receives a 201-response code and fetches results directly in the POST response body.
- If the calculation completes with an error, a user receives a 200-response code and receives error information in the response body.

### 7.1 Setting X-FactSet-Api-Long-Running-Deadline header Via the .NET SDK

To set the X-FactSet-Api-Long-Running-Deadline header and specify the period that will result in an interactive endpoint response, you can do the following:

- Create an Integer `xFactSetApiLongRunningDeadline` parameter and set to desired amount of time – line 36 below
- Pass `xFactSetApiLongRunningDeadline` parameter in the `PostAndCalculate()` or `PutAndCalculate()` methods – line 43 below
  - Note that all engines calculations APIs (PaCalculationsApi, SparCalculationsApi, VaultCalculationsApi, QuantCalculationsApi, FiCalculationsApi,) support the `xFactSetApiLongRunningDeadline` parameter via the `PostAndCalculate()` and `PutAndCalculate()` endpoints
- In this example, the X-FactSet-Api-Long-Running-Deadline has been set to 10 seconds, meaning that if

the calculation takes less than 10 seconds to compute, it will return a 201 with the results directly (without needing to poll the status and fetch the results in separate calls) and if the calculation is not completed within 10s, it will fetch a 202 and go into the long-running mode.

- User can then poll for the status of the calculation using the GET Status by Id endpoint and get the result using the GET Result by Id endpoint once the status is completed.

```

33
34
35
36 var xFactSetApiLongRunningDeadline = 10; // int? | Long running deadline in seconds when only one unit is passed in the POST body. (optional)
37 var cacheControl = "cacheControl_example"; // string | Standard HTTP header. Accepts max-stale. (optional)
38 var pACalculationParametersRoot = new PACalculationParametersRoot(); // PACalculationParametersRoot | Calculation Parameters (optional)
39
40 try
41 {
42     // Create and Run PA calculation
43     PACalculationsApi.PostAndCalculateResponseWrapper result = apiInstance.PostAndCalculate(xFactSetApiLongRunningDeadline,
44     cacheControl, pACalculationParametersRoot);
45
46     switch (result.StatusCode)
47     {
48     case (HttpStatusCode)200:
49         Console.WriteLine(result.Response200);
50         break;
51     case (HttpStatusCode)201:
52         Console.WriteLine(result.Response201);
53         break;
54     case (HttpStatusCode)202:
55         Console.WriteLine(result.Response202);
56         break;
57     }
58 }
59 catch (ApiException e)
60 {
61     Console.WriteLine("Exception when calling PACalculationsApi.PostAndCalculate: " + e.Message );
62     Console.WriteLine("Status Code: " + e.ErrorCode);
63     Console.WriteLine(e.StackTrace);
64 }
65 }
66 }
67 }
68 }
69 }
70 }
71 }

```

## 7.2 Standalone Request Example

**Standalone Request:**

POST https://api.factset.com/analytics/engines/pa/v3/calculations

**Request Headers:**

Content-Type:  
 application/json Cache-  
 Control: max-stale=0  
 X-FactSet-Api-Long-Running-Deadline: 10  
 WeAuthorization: Basic\*\*\*\*\*  
 Accept: /\*/  
 Host: api.factset.com  
 Accept-Encoding: gzip, deflate, br

Content-Length: 739

**Body:**

```
{
  "data": {
    "1": {
      "componentid": "125872E5D836683A3EB09EF8C73D3E1BE854BE4EF6173BAB4BFBB5752788F4E9",
      "accounts": [
        {
          "id": "BENCH:SP50",
          "holdingsmode": "B&H",
        }
      ],
      "dates": {
        "startdate": "20170101",
        "enddate": "20180101",
        "frequency": "Monthly"
      },
      "groups": [
        {
          "id": "E5DB59F608778844CD784DD5659F65AA8A0A9F63F9E4314B2D92761AF0B1B3D9"
        }
      ],
      "benchmarks": [
        {
          "id": "BENCH:R.1000"
        }
      ],
      "currencyisocode": "USD"
    }
  }
}
```

## 8. Installing v3-Compatible .NET SDK

Follow the instructions outlined [here](#) to install the FactSet Enterprise .NET SDKs

Engine	SDK
PA	<a href="#">FactSet.SDK.PAEngine 1.0.0</a>
SPAR	<a href="#">FactSet.SDK.SPAREngine 1.0.0</a>
Vault	<a href="#">FactSet.SDK.Vault 1.0.0</a>
Quant	<a href="#">FactSet.SDK.QuantEngine 1.0.0</a>
FI	<a href="#">FactSet.SDK.FixedIncomeCalculation 1.0.0</a>

For users currently on version-3 of the APIs and using **fds.analytics.sdk** can migrate to **FactSet.sdk.<Engine>**.

We encourage all existing users on **Analytics SDK v5.0.0** and above, to migrate over to **FactSet Enterprise SDK v1.0.0**.

All upcoming enhancements to the APIs will be supported only on **FactSet Enterprise SDKs**. Analytics SDKs will be put into sunset mode and will only be maintained to address any bugs identified.

## 9. Troubleshooting

The Please file an issue tracker with category “**Analytics APIs – Support**” and sub-category “**v3 Migration**” in case you need any assistance or have any feedback.

Please also note that we are targeting to retire the Analytics SDKs by the end of **August 2024**.

## 10. Appendix

### 10.1 CalculationStatusRoot

Engine	API-Specific Method
PA	<a href="#">CalculationStatusRoot</a>
SPAR	<a href="#">CalculationStatusRoot</a>
Vault	<a href="#">CalculationStatusRoot</a>
Quant	<a href="#">CalculationStatusRoot</a>

### 10.2 ObjectRoot

Engine	API-Specific Method
PA	<a href="#">ObjectRoot</a>
SPAR	<a href="#">ObjectRoot</a>
Vault	<a href="#">ObjectRoot</a>
Quant	<a href="#">ObjectRoot</a>
FI	<a href="#">ObjectRoot</a>