Flintfox Performance Pricing - API

Last Modified on 2025/02/07

Contents

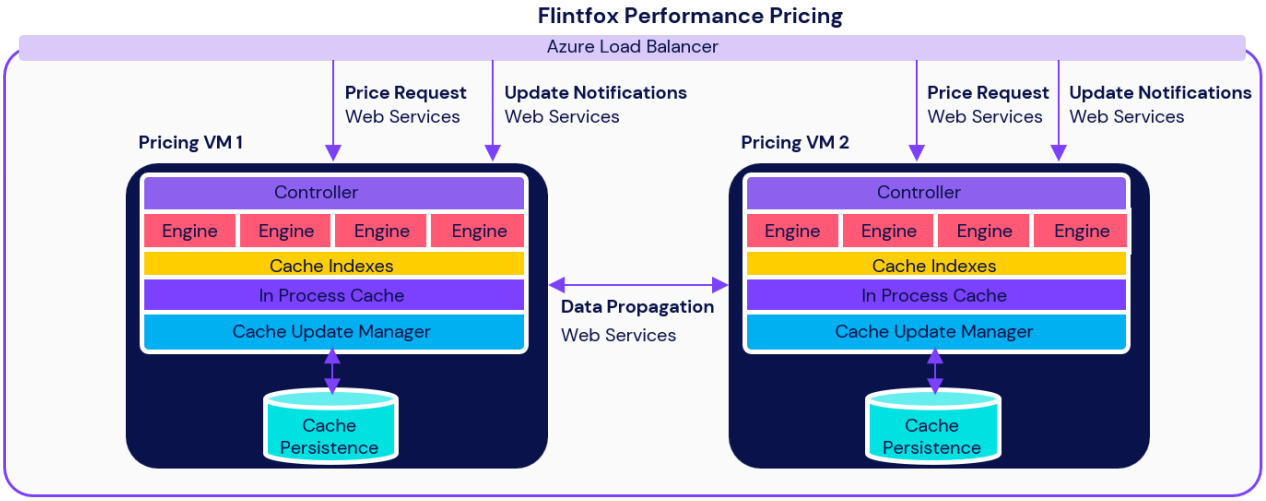
Note:

The product this article relates to was renamed Flintfox (for any ERP) in December 2022, from the earlier RMx. However "under the covers" identification of many tables, fields, folders, scripts, urls, and some processes retain the use of rmx.

Introduction

Flintfox Performance Pricing can meet the functional pricing requirements of enterprise customers and can deliver millisecond response times. The pricing "bandwidth" of the pricing engine allows batch price request processing, with multiple batches executed simultaneously across multiple server cores, while maintaining the same single price request performance.

With the use of an in-memory caching solution high pricing performance is achieved. With an application service consisting of a thread safe local cache and proprietary indexing strategies, along with a thread controller, and pricing engine, efficient use of all available server processing units is achieved for multi-threaded simultaneous executions.



The Flintfox performance pricing engine delivers;

- High performance, in-memory, pricing engine
- Batched processing
- Pre-integrated to Flintfox for D365 and Flintfox for Any ERP
- More nodes can be added for additional bandwidth and scalability

API Security Mode

The Performance Pricing Engine has 3 types of security strategy.

- **None**
  - This will ignore any Authorization token or API key and continue with the requested action
- **ApiKey**
  - This is a Custom request header with the header name => X-API-Key
- **JWT**
  - This will check whether the authorization token is valid or not

© 2009-2022

The security mode is configured in the *appsettings.json* file and be set to any one of the three values listed above.

```
"PricingEngine": {
...
"AuthenticationStrategy": "ApiKey"
...
  }
```

The pricing engine provides a local website to help with configuration of API keys and cannot be accessed remotely. Below is an example of how to access the settings page on the local server configured to listen on port 8005.

-

How to create an API key

Multiple API keys can be created and issued to third party integrators. If the key is removed from the list, it is effectively revoked, and will no longer have permission to access the price engine. This provides the administrator the ability to provide and revoke access to multiple integrators.

- Go to
- There are two ways to generate API key:
  - Generate a Guid API key using the Generate button
  - Manually enter a Guid API key

How to create a message API key

This key is used when there are more than one Pricing Engine running, we need to sync all pricing engine instances' cache when one of the pricing engines is updated.

- Go to
- There are two ways to generate an API key:
  - Generate a Guid API key using the Generate button
  - Manually enter a Guid API key

Initialize Default RMx Pricing Client

DefaultRMxPricingClient is an example wrapper of a Http client that was generated using a Swagger client generation tool. It accepts a HttpClient in the constructor and uses this for all subsequent API calls.

```
var httpClient = new HttpClient {BaseAddress = new Uri(ConfigurationManager.AppSettings.Get("URL"))};

httpClient.DefaultRequestHeaders.Add("X-API-KEY", "apiKey");

httpClient.DefaultRequestHeaders.Authorization = new AuthenticationHeaderValue("Bearer", "token");

_rmxPricingClient = new DefaultRMxPricingClient(httpClient);
```

Swagger UI

NOTE: For swagger UI and the generated swagger.json file to function correctly, go to appsettings.json, find setting PricingEngine:UseUtf8Json and set to false. Set the value back to true once it is done.

Pricing API

The pricing engine exposes two methods to price items; PriceSingle is used to get a single price request result from a single price request parameter and PriceCollection is used to get a collection of price request results from a collection of price request parameters, one for each i.e. batched pricing.
The type Price Request Parameter contains information required to price an order, including pricing date, currency, lines to price. Each line has a type DocumentLine and contains a product together with its fields such as unit of measure, quantity. The same class DocumentLine used to pass Product to RMx Pricing may contain output fields (e.g. list price) as part of the Result class.

Price Single Method

Namespace: RMxPricing

Syntax

```
PriceRequestResult PriceSingle(PriceRequestParameter serviceParameter, string clientDataSource);
```

Parameters

Name	Type	Description
serviceParameter	PriceRequestParameter	The parameter information to calculate a price result for, includes Customer, Product, Date, UOM, Quantity.
clientDataSource	string	Combination of domain, SQL server name and DB name to uniquely identify a data source

ReturnValue

The price result as a PriceRequestResult type for the serviceParameter parameter.

Usage

```
public async Task GetPrice()
{
var requestBody = new PriceRequestParameter
{
      Currency = "NZD",
      DocumentType = DocumentType.SalesOrder,
      CustVend = new CustVend
{
         Code = "CDX-001",
         CompanyKey = 22565420944,
         EntityType = EntityType.Customer
},
      DocumentLines = new List<DocumentLine>
{
new DocumentLine
{
            LineFields = new DocumentLineFields
{
               EffectiveDate = new DateTime(2021, 02, 10, 0, 0, 0),
               LineAction = LineAction.Unspecified,
               LineType = DocumentLineType.StandardLine,
               Product = new Product {Code = "T0003"},
               Quantity = 600,
               UnitOfMeasure = new UnitOfMeasure {UnitOfMeasureCode = "1/2 cu. in"}
}
}
}
};
PriceRequestResult response = await _rmxPricingClient.GetPriceAsync(requestBody);


if (response?.DocumentLines != null && response.DocumentLines.Any())
{
DocumentLineFields lineFields = response.DocumentLines[0].LineFields;
decimal listPrice = lineFields.ListPrice;
decimal netPrice = lineFields.ListPrice;
}
}
```

Postman

Http method	POST
URL	/api/pricingEngine/price/priceSingle
Query parameters	clientDataSource=

Request Body


```
{
"company": "USMF",
"currency": "NZD",
"documentType": "SalesOrder",
"custVend": {
"code": "CDX-001",
"companyKey": 22565420994,
"entityType": "Customer"
   },
"documentLines": [
      {
"lineFields": {
"effectiveDate": "2021-02-10T00:00:00",
"lineAction": "Unspecified",
"lineType": "StandardLine",
"product": {
"code": "T0003"
           },
"quantity": 600,
"unitOfMeasure": {
"unitOfMeasureCode": "1\/2 cu. in"
           }
        }
     }
  ]
}
```


Response Status 200

Response Body


```
{
"identifier": "00000000-0000-0000-0000-000000000000",
"companyKey": 22565420944,
"custVendKey": 246454,
"salesId": null,
"lineFields": null,
"priceModelCode": "QA_Planning",
"priceLevels": null,
"freeGoods": null,
"tradeAgreementsWithOrderUsageLimit": null,
"tAPriceWithQtyUsageLimit": null,
"DocumentLines": [
      {
"lineFields": {
"documentLineSourceId": "3fa841f6-422e-4388-9581-1838546fda3f",
"documentLineKey": null,
"product": {
"productKey": 244897,
"productSourceRecId": 0,
"code": "T0003",
"ecoResProduct": null
           },
"unitOfMeasure": {
"unitOfMeasureKey": 3913,
"unitOfMeasureCode": "1/2 cu. in"
           },
"quantity": 600,
"effectiveDate": "2021-02-10T00:00:00",
"isInvoiced": false,
"lineType": 1,
"manualTradeAgreementVersionKey": null,
```

```
"freeGoodTradeAgreementKey": null,
"costPrice": null,
"listPrice": 999.0000000000,
"netPrice": 999.0000000000,
"netPriceExtended": 599400.0000000000,
"unitDiscount": 0.0000000000,
"extendedDiscount": 0.0000000000,
"roundingAdjustment": 0.0000000000,
"lineReference": null,
"hasOrderBasedFreeGoods": false,
"hasOrderBasedBreak": false,
"hasBreakComponent": false,
"hasManualAgreementAvailable": false,
"hasFreeGoodAvailable": false,
"manualAgreementIsUsed": false,
"shipDate": null,
"pricingQuantity": null,
"disablePricingValidation": false,
"isManualPrice": false,
"manualPriceCategoryKey": null,
"zeroPrice": false,
"lineFieldNameValues": null,
"inventDimFieldNameValues": null,
"lineExtensions": null,
"lineAttributes": null,
"lineAction": 0,
"manualTradeAgreements": null,
"notAppliedPriceBreakKeys": null,
"accrualRequestedPrices": null,
"accrualPricingDetails": null,
"sequenceNumber": 0
        },
"priceLevels": [
          {
"priceModelLevelKey": 4444,
"priceModelLevelCode": "LIST",
"priceModelLevelOrder": 10,
"isListPriceLevel": true,
"isNetPriceLevel": false,
"isHoldingLevel": false,
"isMiscChargeLevel": false,
"isAccrualLevel": false,
"levelType": 1,
"unitAmount": 999.0000000000,
"extendedAmount": 599400.0000000000,
"miscChargeAmount": 0,
"details": [
                {
"tradeAgreementVersionKey": 111633,
"unitPrice": 999.0000000000,
"extendedPrice": 599400.0000000000,
"componentKey": 5637154609,
"categoryKey": 7233,
"disablePricingValidation": false,
"tradeAgreementVersionCode": "TA-000000601",
"componentCode": "List Price",
"categoryCode": "LISTPR",
"priceDetailTag": null,
"runType": null,
"priceKey": 506470,
"priceBreakKey": null,
"componentValue": 999.0000000000,
"priceType": 1
                }
            ]
```

```
            },
            {
"priceModelLevelKey": 4448,
"priceModelLevelCode": "NET",
"priceModelLevelOrder": 30,
"isListPriceLevel": false,
"isNetPriceLevel": true,
"isHoldingLevel": true,
"isMiscChargeLevel": false,
"isAccrualLevel": false,
"levelType": 2,
"unitAmount": 999.0000000000,
"extendedAmount": 599400.0000000000,
"miscChargeAmount": 0,
"details": null
            }
        ]
      }
   ],
"errorMessage": "",
"errorLabelFormatters": [],
"debugInfo": {
"sessionId": 0,
"computerName": null,
"logLevel": 0,
"xmlTrace": null,
"htmlTrace": null
   },
"headerAttributes": null,
"nodeLB": null
}
```


Price Collection Method

Namespace: RMxPricing

Syntax

```
List<PriceRequestResult> PriceCollection(List<PriceRequestParameter> serviceParameters, string clientDataSource);
```

Parameters

Name	Type	Description
ServiceParameters	List<PriceRequestParameter>	A collection of parameters with information to calculate a price result for, includes Customer, Product, Date, UOM, Quantity.
clientDataSource	string	Combination of domain, SQL server name and DB name to uniquely identify a data source

ReturnValue

The price result as a PriceRequestResult type for the serviceParameter parameter.

Usage

```
public async Task PriceCollection()
{
var dataSource = "dataSource";
var requestList = new List<PriceRequestParameter>();
var productCodes = new[] {"T0001", "T0003", "T0004"};


foreach (string productCode in productCodes)
{
var request = new PriceRequestParameter
{
          Currency = "NZD",
         DocumentType = DocumentType.SalesOrder,
          CustVend = new CustVend
{
             Code = "CDX-001",
             CompanyKey = 22565420944,
             EntityType = EntityType.Customer
},
        DocumentLines = new List<DocumentLine>
{
new DocumentLine
{
                LineFields = new DocumentLineFields
{
                    EffectiveDate = new DateTime(2021, 02, 10, 0, 0, 0),
                    LineAction = LineAction.Unspecified,
                    LineType = DocumentLineType.StandardLine,
                    Product = new Product {Code = productCode},
                    Quantity = 600,
                   UnitOfMeasure = new UnitOfMeasure {UnitOfMeasureCode = "1/2 cu. in"}
}
}
}
};
      requestList.Add(request);
}


    _rmxPricingClient = new DefaultRMxPricingClient(httpClient);
IList<PriceRequestResult> response = await _rmxPricingClient.PriceCollectionAsync(dataSource, requestList);
List<PriceRequestResult> validPrices = response != null && response.Any() ?
       response.Where(x => x.DocumentLines.Any()).ToList() :
new List<PriceRequestResult>();


if (validPrices.Any())
{
foreach (PriceRequestResult priceRequestResult in validPrices)
{
DocumentLineFields lineFields = priceRequestResult.DocumentLines[0].LineFields;
decimal listPrice = lineFields.ListPrice;
decimal netPrice = lineFields.ListPrice;
long productId = lineFields.Product.ProductSourceRecId;


PriceRequestParameter parameter = requestList.FirstOrDefault(x => x.Identifier == priceRequestResult.Identifier);
long custVendRecId = parameter?.CustVend?.CustVendSourceRecId ?? 0;
}
}
}
```

Postman

Http method	POST
Url	/api/pricingEngine/price/priceCollection
Query parameters	clientDataSource=

Request Body	[
	{
	"company": "USMF",
	"currency": "NZD",
	"documentType": "SalesOrder",
	"custVend": {
	"code": "CDX-001",
	"companyKey": 22565420994,
	"entityType": "Customer"
	},
	"documentLines": [
	{
	"lineFields": {
	"effectiveDate": "2021-02-10T00:00:00",
	"lineAction": "Unspecified",
	"lineType": "StandardLine",
	"product": {
	"code": "T0003"
	},
	"quantity": 600,
	"unitOfMeasure": {
	"unitOfMeasureCode": "1\/2 cu. in"
	}
	}
	}
	]
	},
	{
	"company": "USMF",
	"currency": "NZD",
	"documentType": "SalesOrder",
	"custVend": {
	"code": "CDX-001",
	"companyKey": 22565420994,
	"entityType": "Customer"
	},
	"documentLines": [
	{
	"lineFields": {
	"effectiveDate": "2021-02-10T00:00:00",
	"lineAction": "Unspecified",
	"lineType": "StandardLine",
	"product": {
	"code": "T0004"
	},
	"quantity": 600,
	"unitOfMeasure": {
	"unitOfMeasureCode": "1\/2 cu. in"
	}
	}
	}
	]
	}
	]

Response Status	200
Response Body	[

```
  {
"identifier": "00000000-0000-0000-0000-000000000000",
"companyKey": 22565420944,
"custVendKey": 246454,
"salesId": null,
"lineFields": null,
"priceModelCode": "QA_Planning",
"priceLevels": null,
"freeGoods": null,
"tradeAgreementsWithOrderUsageLimit": null,
"tAPriceWithQtyUsageLimit": null,
"DocumentLines": [
        {
"lineFields": {
"documentLineSourceId": "7a346c0f-54ff-4377-adce-69e0acecd0c4",
"documentLineKey": null,
"product": {
"productKey": 244897,
"productSourceRecId": 0,
"code": "T0003",
"ecoResProduct": null
            },
"unitOfMeasure": {
"unitOfMeasureKey": 3913,
"unitOfMeasureCode": "1/2 cu. in"
            },
"quantity": 600,
"effectiveDate": "2021-02-10T00:00:00",
"isInvoiced": false,
"lineType": 1,
"manualTradeAgreementVersionKey": null,
"freeGoodTradeAgreementKey": null,
"costPrice": null,
"listPrice": 999.0000000000,
"netPrice": 999.0000000000,
"netPriceExtended": 599400.0000000000,
"unitDiscount": 0.0000000000,
"extendedDiscount": 0.0000000000,
"roundingAdjustment": 0.0000000000,
"lineReference": null,
"hasOrderBasedFreeGoods": false,
"hasOrderBasedBreak": false,
"hasBreakComponent": false,
"hasManualAgreementAvailable": false,
"hasFreeGoodAvailable": false,
"manualAgreementIsUsed": false,
"shipDate": null,
"pricingQuantity": null,
"disablePricingValidation": false,
"isManualPrice": false,
"manualPriceCategoryKey": null,
"zeroPrice": false,
"lineFieldNameValues": null,
"inventDimFieldNameValues": null,
"lineExtensions": null,
"lineAttributes": null,
"lineAction": 0,
"manualTradeAgreements": null,
"notAppliedPriceBreakKeys": null,
"accrualRequestedPrices": null,
"accrualPricingDetails": null,
"sequenceNumber": 0
        },
"priceLevels": [
```

```
            {
"priceModelLevelKey": 4444,
"priceModelLevelCode": "LIST",
"priceModelLevelOrder": 10,
"isListPriceLevel": true,
"isNetPriceLevel": false,
"isHoldingLevel": false,
"isMiscChargeLevel": false,
"isAccrualLevel": false,
"levelType": 1,
"unitAmount": 999.0000000000,
"extendedAmount": 599400.0000000000,
"miscChargeAmount": 0,
"details": [
                {
"tradeAgreementVersionKey": 111633,
"unitPrice": 999.0000000000,
"extendedPrice": 599400.0000000000,
"componentKey": 5637154609,
"categoryKey": 7233,
"disablePricingValidation": false,
"tradeAgreementVersionCode": "TA-000000601",
"componentCode": "List Price",
"categoryCode": "LISTPR",
"priceDetailTag": null,
"runType": null,
"priceKey": 506470,
"priceBreakKey": null,
"componentValue": 999.0000000000,
"priceType": 1
                }
              ]
            },
            {
"priceModelLevelKey": 4448,
"priceModelLevelCode": "NET",
"priceModelLevelOrder": 30,
"isListPriceLevel": false,
"isNetPriceLevel": true,
"isHoldingLevel": true,
"isMiscChargeLevel": false,
"isAccrualLevel": false,
"levelType": 2,
"unitAmount": 999.0000000000,
"extendedAmount": 599400.0000000000,
"miscChargeAmount": 0,
"details": null
            }
          ]
        }
      ],
"errorMessage": "",
"errorLabelFormatters": [],
"debugInfo": {
"sessionId": 0,
"computerName": null,
"logLevel": 0,
"xmlTrace": null,
"htmlTrace": null
      },
"headerAttributes": null,
"nodeLB": null
   },
   {
"identifier": "00000000-0000-0000-0000-000000000000",
```

```
"companyKey": 22565420944,
"custVendKey": 246454,
"salesId": null,
"lineFields": null,
"priceModelCode": "QA_Planning",
"priceLevels": null,
"freeGoods": null,
"tradeAgreementsWithOrderUsageLimit": null,
"tAPriceWithQtyUsageLimit": null,
"DocumentLines": [
        {
"lineFields": {
"documentLineSourceId": "24b0ea29-3897-4ddd-858d-5c3e9cefd571",
"documentLineKey": null,
"product": {
"productKey": 246246,
"productSourceRecId": 0,
"code": "T0004",
"ecoResProduct": null
            },
"unitOfMeasure": {
"unitOfMeasureKey": 3913,
"unitOfMeasureCode": "1/2 cu. in"
            },
"quantity": 600,
"effectiveDate": "2021-02-10T00:00:00",
"isInvoiced": false,
"lineType": 1,
"manualTradeAgreementVersionKey": null,
"freeGoodTradeAgreementKey": null,
"costPrice": null,
"listPrice": 999.0000000000,
"netPrice": 999.0000000000,
"netPriceExtended": 599400.0000000000,
"unitDiscount": 0.0000000000,
"extendedDiscount": 0.0000000000,
"roundingAdjustment": 0.0000000000,
"lineReference": null,
"hasOrderBasedFreeGoods": false,
"hasOrderBasedBreak": false,
"hasBreakComponent": false,
"hasManualAgreementAvailable": false,
"hasFreeGoodAvailable": false,
"manualAgreementIsUsed": false,
"shipDate": null,
"pricingQuantity": null,
"disablePricingValidation": false,
"isManualPrice": false,
"manualPriceCategoryKey": null,
"zeroPrice": false,
"lineFieldNameValues": null,
"inventDimFieldNameValues": null,
"lineExtensions": null,
"lineAttributes": null,
"lineAction": 0,
"manualTradeAgreements": null,
"notAppliedPriceBreakKeys": null,
"accrualRequestedPrices": null,
"accrualPricingDetails": null,
"sequenceNumber": 0
        },
"priceLevels": [
            {
"priceModelLevelKey": 4444,
"priceModelLevelCode": "LIST",
```

```
"priceModelLevelOrder": 10,
"isListPriceLevel": true,
"isNetPriceLevel": false,
"isHoldingLevel": false,
"isMiscChargeLevel": false,
"isAccrualLevel": false,
"levelType": 1,
"unitAmount": 999.0000000000,
"extendedAmount": 599400.0000000000,
"miscChargeAmount": 0,
"details": [
                    {
"tradeAgreementVersionKey": 111633,
"unitPrice": 999.0000000000,
"extendedPrice": 599400.0000000000,
"componentKey": 5637154609,
"categoryKey": 7233,
"disablePricingValidation": false,
"tradeAgreementVersionCode": "TA-000000601",
"componentCode": "List Price",
"categoryCode": "LISTPR",
"priceDetailTag": null,
"runType": null,
"priceKey": 506470,
"priceBreakKey": null,
"componentValue": 999.0000000000,
"priceType": 1
                    }
                ]
            },
            {
"priceModelLevelKey": 4448,
"priceModelLevelCode": "NET",
"priceModelLevelOrder": 30,
"isListPriceLevel": false,
"isNetPriceLevel": true,
"isHoldingLevel": true,
"isMiscChargeLevel": false,
"isAccrualLevel": false,
"levelType": 2,
"unitAmount": 999.0000000000,
"extendedAmount": 599400.0000000000,
"miscChargeAmount": 0,
"details": null
            }
        ]
    }
  ],
"errorMessage": "",
"errorLabelFormatters": [],
"debugInfo": {
"sessionId": 0,
"computerName": null,
"logLevel": 0,
"xmlTrace": null,
"htmlTrace": null
  },
"headerAttributes": null,
"nodeLB": null
 }
]
```

Price Request Parameter Class

Syntax

public class PriceRequestParameter

```
{
  public Guid Identifier { get; set; }
  public CustVend CustVend { get; set; }
  public Guid DocumentSourceId { get; set; }
  public string SalesId { get; set; }
  public string Currency { get; set; }
  public DocumentType DocumentType { get; set; }
  public DocumentLineFields LineFields { get; set; }
  public List<DocumentLine> DocumentLines { get; set; }
  public List<FieldNameValue> HeaderFieldNameValues { get; set; }
  public List<OrderUsage> TradeAgreementsWithOrderUsageLimit { get; set; }
  public DebugInformation DebugInfo { get; set; }
  public List<FieldNameValue> HeaderExtensions { get; set; }
  public int? RoundingForExtendedPrice { get; set; }
  public string Language { get; set; }
  public string Company { get; set; }
  public string HeaderAttributes { get; set; }
  public RequestType RequestType { get; set; }
  public AccrualType AccrualType { get; set; }
  public string RunType { get; set; }
  public bool InIcudePlanningSet { get; set; }
}
```

Properties

Name	Type
Identifier	A unique identifier for the parameter. This will match the Identifier property of the returned resultant PriceRequestResult class. When pricing in batches, i.e. using PriceRequest method and passing a collection of parameters the Identifier can be used to match the returned collection of results with the parameters.
CustVend	The customer or vendor to be priced.
DocumentSourceId	The RecId/Key of the document being priced. Optional.
SalesId	A user defined, human recognisable, identifier for the document being priced. This will match the SalesId of the result. Optional.
Currency	The currency code for this price request.
DocumentType	The type of document being priced. Use SalesOrder for non-document customer pricing and PurchaseOrder for non-document vendor pricing.
LineFields	The order line information for the item being priced. Optional if DocumentLines are provided.
DocumentLines	The document line(s) with product information to be priced.
HeaderFieldNameValues	Used for Transaction pricing. HeaderFieldNameValues will have either the whole order-level record (like SalesTable) passed as FieldName-FieldValue pairs, or preferably, a list of only those FieldName-FieldValue pairs which MAY contribute to transaction pricing. Potential candidates are those referenced in Flintfox.ExternalPriceRequestMapping for Level = Header
TradeAgreementsWithOrderUsageLimit	The Trade Agreements with usage data where Usage Limits have been specified.
DebugInfo	Information used for data logging (for tracing/debugging)
HeaderExtensions	For passing additional name-value pairs for the purpose of customization
RoundingForExtendedPrice	Number of decimal places to use for extended price rounding
Language	Specifies preferred language for diagnostic messages
Company	Company code

HeaderAttributes	When RequestType is PriceAccrualPlanned then HeaderAttributes is an output parameter containing serialized collections of HeaderNameValuePairs and HeaderExtensions. When RequestType is AccrualActual then HeaderAttributes is an input parameter used to calculate accrual details.
RequestType	A type of pricing request which can takes values of PriceOnly (standard pricing), PriceAccrualPlanned (standard pricing and accrual pricing details) and AccrualActual (only accrual details for a given accrual RunType)
AccrualType	A type of accrual (Incremental, FirstPass, SecondPass). Only required for the RequestType=AccrualActual.
RunType	Accrual RunType for accrual calculatinos
IncludePlanningSet	When set to TRUE this flag allows trade agreements in Planning state to be used in pricing together with the Live, Finished etc trade agreements. When set to FALSE then only those agreements, which have a LIVE trade agreement set, can be used in pricing. This feature is currently supported in Flintfox for any ERP, but not in Flintfox for Dynamics 365

Cust Vend Class

Syntax

```
public
 class CustVend
{
  public long CustVendKey { get; set; }
  public long CustVendSourceRecId { get; set; }
  public long CompanyKey { get; set; }
  public EntityType EntityType { get; set; }
public string Code { get; set; }
}
```

Properties

Name	Type
CustVendKey	The RecId/Key of the Flintfox customer/vendor entity record. Not required if CustVendSourceRecId is specified.
CustVendSourceRecId	The RecId/Key of the ERP customer/vendor record. Not required if the CustVendKey is specified.
CompanyKey	RecId of the Flintfox Company record customer/vendor belongs to. Only required if Code is provided while CustVendKey and CustVendSourceRecId are not.
EntityType	Customer or Vendor
Code	Customer or Vendor Code. Not required if the CustVendKey or CustVendSourceRecId are specified.

Product Class

Syntax

```
public class Product
{
  public long ProductKey { get; set; }
  public long ProductSourceRecId { get; set; }
public string Code { get; set; }
}
```

Properties

Name	Type
ProductKey	The RecId/Key of the Flintfox product entity record. Not required if ProductSourceRecId is specified.
ProductSourceRecId	The RecId/Key of the ERP product record. Not required if ProductKey is specified.
Code	Product code. Not required if ProductKey is specified.

Unit Of Measure Class

Syntax

```
public class UnitOfMeasure
{
  public long? UnitOfMeasureKey { get; set; }
  public string UnitOfMeasureCode { get; set; }
}
```

Properties

Name	Type
UnitOfMeasureKey	The RecId/Key of the ERP unit of measure record. Not required if UnitOfMeasureCode is specified.
UnitOfMeasureCode	The unit of measure code identified for the unit of measure record. Not required if UnitOfMeasureKey is specified.

Free Good Class

Syntax

```
public class FreeGood
{
  public long TradeAgreementVersionKey { get; set; }
  public long TradeAgreementPriceKey{ get; set; }
  public long TradeAgreementPriceBreakKey{ get; set; }
  public Product RewardProduct { get; set; }
  public UnitOfMeasure UnitOfMeasure { get; set; }
  public decimal Quantity { get; set; }
}
```

Properties

Name	Type

TradeAgreementVersionKey	The Trade Agreement version RecId/Key that contained the Free Good component and price.
TradeAgreementPriceKey	The Trade Agreement Price RecId/Key associated with manually selected Free Good price.
TradeAgreementPriceBreakKey	The Trade Agreement Price Break RecId/Key associated with manually selected Free Good price break.
RewardProduct	The Free Good product.
UnitOfMeasure	The unit of measure of the free good.
Quantity	The quantity of the free good.

Manual Price Class

Syntax

```
public class ManualPrice
{
  public long TradeAgreementPriceKey{ get; set; }
  public long TradeAgreementPriceBreakKey{ get; set; }
  public List<ManualFreeGoods>  ManualFreeGoods { get; set; }
}
```

Properties

Name	Type
TradeAgreementPriceKey	The Trade Agreement Price RecId/Key associated with manually selected Free Good price.
TradeAgreementPriceBreakKey	The Trade Agreement Price Break RecId/Key associated with manually selected Free Good price break.
ManualFreeGoods	Collection of manually selected Free Good products.

Manual Free Good Class

Syntax

```
public class ManualFreeGood
{
  public Product RewardProduct { get; set; }
  public UnitOfMeasure UnitOfMeasure { get; set; }
  public decimal Quantity { get; set; }
}
```

Properties

Name	Type
RewardProduct	The manually selected Free Good product.
UnitOfMeasure	The unit of measure of the free good.
Quantity	The quantity of the free good.

Field Name Value Class

Syntax

```
public class FieldNameValue
{
  public string Name { get; set; }
  public string Value { get; set; }
}
```

Properties

Name	Type
Name	Field name.
Value	Field value.

Order Usage Class

Syntax

```
public class OrderUsage
{
  public long TradeAgreementVersionKey { get; set; }
  public int? OrderUsageCount { get; set; }
}
```

Properties

Name	Type
TradeAgreementVersionKey	The Trade Agreement version RecId/Key
OrderUsageCount	The number of times of the Trade Agreement has been used to date. Used for limiting Trade Agreement inclusion in pricing when Order Usage has been set.

Document Line Class

Syntax

```
public class DocumentLine
{
  public DocumentLineFields LineFields { get; set; }
  public List<PriceRequestResultLevel> PriceLevels { get; set; }
}
```

Properties

Name	Type
LineFields **DocumentLineFields**	The order line information for the item being priced
PriceLevels	The price level information that provides a break down of all the discounts resolved during pricing.

Document Line Fields Class

Syntax

```
public class DocumentLineFields
{
  public Guid DocumentLineSourceId { get; set; }
  public Product Product { get; set; }
  public UnitOfMeasure UnitOfMeasure { get; set; }
  public decimal Quantity { get; set; }
  public DateTime EffectiveDate { get; set; }
  public bool IsInvoiced { get; set; }
  public DocumentLineType LineType { get; set; }
  public long? ManualTradeAgreementVersionKey { get; set; }
  public long? FreeGoodTradeAgreementKey { get; set; }
  public decimal? CostPrice { get; set; }
  public decimal ListPrice { get; set; }
  public decimal NetPrice { get; set; }
  public decimal NetPriceExtended { get; set; }
  public decimal UnitDiscount { get; set; }
  public decimal ExtendedDiscount { get; set; }
  public decimal RoundingAdjustment { get; set; }
  public string LineReference { get; set; }
  public bool HasOrderBasedFreeGoods { get; set; }
  public bool HasOrderBasedBreak { get; set; }
  public bool HasBreakComponent { get; set; }
  public bool HasManualAgreementAvailable { get; set; }
  public bool HasFreeGoodAvailable { get; set; }
  public bool ManualAgreementIsUsed { get; set; }
  public DateTime? ShipDate { get; set; }
  public decimal? PricingQuantity { get; set; }
  public bool DisablePricingValidation { get; set; }
  public bool IsManualPrice { get; set; }
  public long? ManualPriceCategoryKey { get; set; }
  public bool ZeroPrice { get; set; }
  public string LineAttributes { get; set; }
  public LineAction LineAction { get; set; }
  public List<FieldNameValue> LineFieldNameValues { get; set; }
  public List<FieldNameValue> InventDimFieldNameValues { get; set; }
  public List<FieldNameValue> LineExtensions { get; set; }
  public List<ManualTradeAgreement> ManualTradeAgreements { get; set; }
  public List<long> NotAppliedPriceBreakKeys{ get; set; }
  public Dictionary<long, AccrualRequestedPrice> AccrualRequestedPrices { get; set; }
  public string AccrualPricingDetails { get; set; }
}
```

Properties

Name	Type
DocumentLineSourceId	The GUID to uniqutely identify the line.
Product	The product of the line.
UnitOfMeasure	The UOM of the line.
Quantity	The quantity for calculating extended discounts and breaks (if no pricing quantity specified)
EffectiveDate	Line date.
IsInvoiced	True if the line is a sales line and has been invoiced.
LineType	The Line Type of the line. Normally set to StandardLine.

ManualTradeAgreementVersionKey	The RecId/Key of the manual Trade Agreement that is selected to be included in the pricing for this line.
FreeGoodTradeAgreementKey	The RecId/Key of the Trade Agreement containing the Free Good component. Empty if no free goods have been priced relating to this line.
CostPrice	The cost price for the line. Optionally set if CostPrice components are used.
ListPrice	The calculated list price based on the defined List Price level. Usually the first level but not always.
NetPrice	The calculated net price (List – UnitDiscount) for the line
NetPriceExtended	Calculated as NetPrice * Qty
UnitDiscount	The calculated unit discount (aggregate) for the line
ExtendedDiscount	Calculated as UnitDiscount * Qty
RoundingAdjustment	Adjustment due to rounding of extended price.
LineReference	User defined line reference
HasOrderBasedFreeGood	True if an order based free good break is available. A full order re-price may be required.
HasOrderBasedBreak	True if an order based break is available. A full order re-price may be required.
HasBreakComponent	True if a break component is used during price calculation.
HasManualAgreementAvailabler	True if Manual Trade Agreements are available.
HasFreeGoodAvailable	True if Free Goods are available to be selected.
ManualAgreementIsUsed	True if manual Trade Agreements have been selected and used during pricing.
ShipDate	Date of order line shipment. Used with pricing date is set to the Ship Date Type.
PricingQuantity	The quantity used to calculate price breaks. If empty will use Quantity property.
DisablePricingValidation	Set to True if manual Agreement price break validation should be disabled/by-passed.
IsManualPrice	Reserved for future or internal use.
ManualPriceCategoryKey	If a Net Price Override has been defined this is the RecId/Key of the Net Price discount Category.
ZeroPrice	Reserved for future or internal use.
LineAttributes	When RequestType is PriceAccrualPlanned then LinesAttributes is an output parameter containing serialized collections of LineNameValuePairs, InventDimNameValuePairs and LineExtensions as well as calculated pricing details. When RequestType is AccrualActual then LineAttributes is an input parameter used to calculate accrual details.
LineAction	For a manually selected FreeGood line, this can be changed from Create or Update to Delete in course of FreeGood pricing, if it doesn't pass the validation. If this field was equal to Delete initially, then the pricing logic of such a FreeGood line will not be executed.
ManualTradeAgreements	Used for Manual Pricing and Manual Free Goods. It contains a collection of Trade Agreements, Trade Agreement Prices, optionally Price Breaks and Free Good rewards manually selected by an operator.
NotAppliedPriceBreakKeys	Used for Manual Pricing and Manual Free Goods as an output parameter. It contains a collection of those manually selected Price Breaks, which for any reason were not ultimately applied in course of pricing.
LineFieldNameValues	Used for Transaction pricing: LineNameValues will have either the whole line-level record (like SalesLine) passed as FieldName-FieldValue pairs, or preferably, a list of only those FieldName-FieldValue pairs which MAY contribute to transaction pricing. Potential candidates are those referenced in Flintfox.ExternalPriceRequestMapping for Level = Line
InventDimFieldNameValues	Similar to LineFieldNameValues, only for InventDim record and Level = InventDim
LineExtensions	For passing additional name-value pairs for the purpose of customization
AccrualRequestedPrices	In accrual calculations on the First and Second pass this field may be populated for those prices which required adjustments (e.g. if they originate from "AutoAdjust" trade agreements)
AccrualPricingDetails	For RequestType = PriceAccrualPlanned this is an output parameter. It contains serialized Pricing Details to be saved by ERP and later used in accrual calculations. For RequestType = AccrualActual this is an input parameter with previously saved Pricing Details.

Price Request Result Class

The class that returns all the results of a nPriceRequest based on a PriceRequestParameter.
Namespace: Flintfox.nPrice.Contract

Syntax

```
public class PriceRequestResult
{
  public Guid Identifier { get; set; }
  public long CompanyKey { get; set; }
public long CustVendKey { get; set; }
  public string SalesId { get; set; }
  public DocumentLineFields LineFields { get; set; }
  public List<PriceRequestResultLevel> PriceLevels { get; set; }
  public List<FreeGood> FreeGoods { get; set; }
  public List<OrderUsage> TradeAgreementsWithOrderUsageLimit { get; set; }
  public List<DocumentLine> DocumentLines { get; set; }
  public string ErrorMessage { get; set; }
  public List<ErrorLabelFormatter> ErrorLabelFormatters { get; set; }
  public DebugInformation DebugInfo { get; set; }
  public string PriceModelCode { get; set; }
  public string HeaderAttributes { get; set; }
}
```

Properties

Name	Type	Description
Identifier	Guid	The GUID of the price result that corresponds to the GUID of the parameter. Used to match multiple results with multiple paramters when issued as a single request i.e. batch pricing.
CompanyKey	long	DataAreaId of AX Legal Entity for the priced order/request
CustVendKey	long	RecId of Flintfox Entity table for Customer/Vendor
SalesId	string	The SalesID of the PriceRequestParameter
LineFields	DocumentLineFields	Only populated if corresponding PriceRequestParameter.LineFields were provided.
PriceLevels	List<PriceRequestResultLevel>	Containes detailed pricing information with discounts split per price level and then to details including the Trade Agreement, Component and Category of the discount.
FreeGoods	List<FreeGood>	This collection is only provided if FreeGoods were applied in course of pricing
TradeAgreementsWithOrderUsageLimit	List<OrderUsage>	Provided if trade agreements with OrderUsageLimit were applied in course of pricing.
DocumentLines	List<DocumentLine>	The pricing results of the lines that were in the PriceRequestParameter
ErrorMessage	String	Any error messages that occurred during pricing. This will be empty if pricing was successful.
DebugInfo	DebugInformation	Debug information if LogLevel is set to return any. This will contain information related to the calculation of pricing.
PriceModelCode	String	Price model actually used in pricing
HeaderAttributes	String	When RequestType is PriceAccrualPlanned then HeaderAttributes is an output parameter containing serialized collections of HeaderNameValuePairs and HeaderExtensions.

ErrorLabelFormatters	**List<ErrorLabelFormatter>**	Used to pass back to the caller an error ID collection to format an error message according to the caller's rules

Price Request Result Level Class

Syntax

```
public class PriceRequestResultLevel
{
  public long PriceModelLevelKey { get; set; }
  public string PriceModelLevelCode { get; set; }
  public int PriceModelLevelOrder { get; set; }
  public bool IsListPriceLevel { get; set; }
  public bool IsNetPriceLevel { get; set; }
  public bool IsHoldingLevel { get; set; }
  public bool IsMiscChargeLevel { get; set; }
  public decimal UnitAmount { get; set; }
  public decimal ExtendedAmount { get; set; }
  public decimal MiscChargeAmount { get; set; }
  public List<PriceRequestResultDetail> Details { get; set; }
}
```

Properties

Name	Type
PriceModelLevelKey	The RecId/Key of the Price Model Level.
PriceModelLevelCode	The Code identifier of the Price Model Level.
PriceModelLevelOrder	The Order of the Price Model Level.
IsListPriceLevel	True if the level has been defined as the List Price level.
IsNetPriceLevel	True if the level has been defined as the Net Price level.
IsHoldingLevel	True if the level has been defined as a Holding level.
IsMiscChargeLevel	True if the level is a Miscellaneous Charges level.
UnitAmount	Running Total
ExtendedAmount	simply UnitAmount * Qty
MiscChargeAmount	Misc charge = Sum(priceDetails)
Details	The price details of the level containing the break down of individual price discounts along with Trade Agreement, Component and Category.

Price Request Result Detail Class

Syntax

```
public class PriceRequestResultDetail
{
  public long? TradeAgreementVersionKey { get; set; }
  public decimal UnitPrice { get; set; }
  public decimal ExtendedPrice { get; set; }  // simply UnitPrice * Qty
  public long? ComponentKey { get; set; }
  public long CategoryKey { get; set; }
  public bool DisablePricingValidation { get; set; }
public string TradeAgreementVersionCode { get; set; }
public string ComponentCode { get; set; }
public string CategoryCode { get; set; }
public string PriceDetailTag { get; set; }
public string RunType { get; set; }
public long? PriceKey { get; set; }
}
```

Properties

Name	Type
TradeAgreementVersionKey	The RecId/Key of the Trade Agreement version where the price originated from.
UnitPrice	The Unit Price of the price/discount.
ExtendedPrice	The Extended Price (Unit Price * Quantity) of the discount/price.
ComponentKey	The RecId/Key of the Component for the price.
CategoryKey	The RecId/Key fo the Category for the price.
DisablePricingValidation	True if the Trade Agreement break trigger requirements were disabled.
TradeAgreementVersionCode	Trade Agreement version, which provided the current price record.
ComponentCode	Component code corresponding to the current price record.
CategoryCode	Category code corresponding to the current trade agreement component.
PriceDetailTag	Reserved for future use and customization to pass any supplementary information.
RunType	In case of accrual calculations, this field shows the run type of the used trade agreement.
PriceKey	Record ID corresponding to the current price record.

Debug Information Class

Syntax

```
public class DebugInformation
{
    public int SessionId { get; set; }
    public string ComputerName { get; set; }
    public LogLevel LogLevel { get; set; }
    public string XmlTrace { get; set; }
    public string HtmlTrace { get; set; }
}
```

Properties

Name	Type
SessionId	Reserved for future or internal use.
ComputerName	Reserved for future or internal use.
LogLevel	The logging level/detail for retrieving pricing debug information.
XmlTrace	The pricing debug information in XML format. Can be translated using XSLT to HTML format as required.
HtmlTrace	The pricing debug information in a pre-formatted HTML format.

Error Label Formatter Class

Syntax

```
public class ErrorLabelFormatter
{
  public int ErrorId { get; set; }
  public List<string> Arguments { get; set; }
}
```

Properties

Name	Type
ErrorId	Integer field to denote error ID as it's know to RMx. Used to format error messages on the caller side according to caller rules.
Arguments	List of arguments for error messages.

Accrual Requested Price Class

Syntax

```
public class AccrualRequestedPrice
{
  public BreakRule? BreakRule { get; set; }
  public bool AdjustAgreement { get; set; }
  public bool IsValid { get; set; }
}
```

Properties

Name	Type
BreakRule	For a price corresponding to an accrual component with breaks this parameter is set to the BreakRule of the corresponding component (Inclusive, Stepped break)
AdjustAgreement	For accrual trade agreements which require adjustment (e.g. "AutoAdjust" agreements) this parameter shall be set to true.
IsValid	On a FirstPass of accrual run this field is an output parameter indicating if the price is applicable.

Document Line Type Enumeration

Syntax

```
public enum DocumentLineType
{
  Unspecified = 0,
  StandardLine = 1,
  FreeGoodLine = 2
}
```

Document Type Enumeration

Syntax

```
public enum DocumentType
{
  Unspecified = 0,
  SalesInvoice = 1,
  SalesOrder = 2,
  SalesShipment = 3,
  SalesTrace = 4,
  PriceRequestResult = 5,
  PurchaseOrder = 6,
  PurchaseInvoice = 7,
  PurchaseTran = 8,
  PurchaseReceipt = 9,
  PurchasePacking = 10,
  SalesQuotation = 11,
  SalesPacking = 12
}
```

Entity Type Enumeration

Syntax

```
public enum EntityType
{
  Unspecified = 0,
  Customer = 1,
  Product = 2,
  Attribute = 3,
  Vendor = 4,
  Transaction = 5
}
```

Log Level Enumeration

Syntax

```
public enum LogLevel
{
  Unspecified = 0,
  Information = 1,
  Error = 2,
  Trace = 3,
  Debug = 4,
  Verbose = 5
}
```

Request Type Enumeration

Syntax

```
public enum RequestType
{
    PriceOnly = 0,
    PriceAccrualPlanned = 1,
    AccrualActual = 2
}
```

Accrual Type Enumeration

Syntax

```
public enum AccrualType
{
    Unspecified = 0,
    Incremental = 1,
    FirstPass = 2,
    SecondPass = 3
}
```

Line Action Enumeration

Syntax

```
public enum LineAction
{
    Unspecified = 0,
    Create = 1,
    Update = 2,
    Delete = 3,
    None = 4
}
```

Break Rule Enumeration

Syntax

```
public enum BreakRule
{
    Unspecified = 0,
    Stepped = 1,
    Inclusive = 2
}
```

Run Level Enumeration

Syntax

```
public enum RunLevel
{
   Incremental = 0,
   Full = 1
}
```

Cache API

RMx Pricing service supports two different strategies of updating data in its cache. The first one contains methods passing data change notifications rather than serialized data. Upon receiving the data change notification the Pricing Service reads the modified data from the database and places the new data to the cache, possibly replacing the old data. The second strategy comprises methods for receiving and processing modified data in serialized form, thus eliminating the need to ever contact database directly. A running instance of RMx Pricing cannot mix the strategies: it can be configured to use only one of them.

RMx Pricing service exposes a method GetCacheInfo() to query its cache state, returning a cache version, time of most recent synchronization, possibly errors.

Get Cache Info Method

Namespace: RMxPricing

Syntax

```
CacheInfo GetCacheInfo(string companyCode, string clientDataSource);
```

Parameters

Name	Type	Description

companyCode	string	Company code for the queried cache. Applicable to a configuration of Pricing service where data is stored in separate collection sets for each company. In case of storing data for all the companies in the same set of collections this parameter shall be empty.
clientDataSource	string	Combination of domain, SQL server name and DB name to uniquely identify a data source

ReturnValue

A CacheInfo object containing cache version, synchronization status, synchronization time and possibly an error message.

Usage

```
public async Task GetCacheInfo()
```

```
{
var dataSource = "dataSource";
CacheInfo response = await _rmxPricingClient.GetCacheInfoAsync("USMF", dataSource);
}
```

Postman

Http method	GET
URL	/api/pricingEngine/cache/getCacheInfo
Query parameters	companyCode= clientDataSource=
Request Body	none
Response Status	200

Response Body	{
	"isLatest": **true**,
	"version": 0,
	"syncDateTime": "2021-07-25 16:14:53Z",
	"error": "",
	"node": **null**
	}

Cache Info Class

Syntax

```
public class CacheInfo
{
  public int Version{ get; set; }
  public string SyncDateTime { get; set; }
  public bool IsLatest { get; set; }
  public string Error { get; set; }
}
```

Properties

Name	Type
Version	A parameter showing the version of the cache
SyncDateTime	Full synchronization date and time
IsLatest	If full synchronization is complete and there were no error generated in course of synchronization, then this parameter is returned as true, otherwise it is returned as false.
Error	Error message generated in course of processing the data message.

Notification API

RMx Pricing service supports two different strategies of updating data in its cache. The first one contains methods passing data change notifications rather than serialized data. Upon receiving the data change notification, the Pricing Service reads the modified data from the database and places the new data to the cache, possibly replacing the old data.

Add Change Notification Single Method

Namespace: RMxPricing

Syntax

```
string AddChangeNotificationSingle (ChangeNotifyMessage message, string clientDataSource);
```

Parameters

Name	Type	Description
message	ChangeNotifyMessage	Contains information required to query database for a single modified row.
clientDataSource	string	Combination of domain, SQL server name and DB name to uniquely identify a data source

ReturnValue

A string which may contain an error message.

Usage

```
public async Task AddChangeNotificationSingle()
{
var dataSource = "dataSource";
var requestBody = new ChangeNotifyMessage
{
      CacheObjectType = CacheObjectType.TradeAgreementVersion,
      ChangeType = PersistType.Update,
      PrimaryKey = 110846,
      EntityType = EntityType.Unspecified
};
string response = await _rmxPricingClient.AddChangeNotificationSingleAsync(dataSource, requestBody);
}
```

Postman

Http method	POST
URL	/api/pricingEngine/notification/addChangeNotificationSingle
Query parameters	clientDataSource=

Request Body	{
	"cacheObjectType": "TradeAgreementVersion",
	"changeType": "Update",
	"primaryKeys": 5637155828,
	"EntityType": "Unspecified"
	}

Response Status	200
Response Body	""

Add Change Notification Collection Method

Namespace: RMxPricing

Syntax

```
string AddChangeNotificationCollection(List<ChangeNotifyMessage> messages, string clientDataSource);
```

Parameters

Name	Type	Description
messages	List<ChangeNotifyMessage>	Contains information required to query database for a collection of modified rows.
clientDataSource	string	Combination of domain, SQL server name and DB name to uniquely identify a data source

ReturnValue

A string which may contain an error message.

Usage

```
public async Task AddChangeNotificationCollection()
{
var dataSource = "dataSource";
var requestBody1 = new ChangeNotifyMessage
{
       CacheObjectType = CacheObjectType.TradeAgreementVersion,
      ChangeType = PersistType.Update,
      PrimaryKey = 110846,
      EntityType = EntityType.Unspecified
};


var requestBody2 = new ChangeNotifyMessage
{
       CacheObjectType = CacheObjectType.TradeAgreementVersion,
      ChangeType = PersistType.Update,
      PrimaryKey = 110847,
      EntityType = EntityType.Unspecified
};
string response = await _rmxPricingClient.AddChangeNotificationCollectionAsync(dataSource, requestList);
}
```

Postman

Http method	POST
URL	/api/pricingEngine/notification/addChangeNotificationCollection
Query parameters	clientDataSource=

Request Body	[
	{
	"cacheObjectType": "TradeAgreementVersion",
	"changeType": "Update",
	"primaryKeys": 5637155828,
	"EntityType": "Unspecified"
	},
	{
	"cacheObjectType": "TradeAgreementVersion",
	"changeType": "Update",
	"primaryKeys": 5637155827,
	"EntityType": "Unspecified"
	}
	]

Response Status	200
Response Body	""

Change Notify Message Class

Syntax

```
public class ChangeNotifyMessage
{
  public CacheObjectType CacheObjectType { get; set; }
  public PersistType ChangeType{ get; set; }
  public long PrimaryKey { get; set; }
  public long? SourceRecId { get; set; }
  public EntityType EntityType{ get; set; }
  public string Code { get; set; }
  public long? CompanyKey { get; set; }
  public DateTime? TimeStamp{ get; set; }
}
```

Properties

Name	Type
CacheObjectType	A parameter identifying the object type and a database table (or tables) the changed object originates from.
ChangeType	The enumeration denoting the type of object modification (Created, Updated, Deleted).
PrimaryKey	The primary key of a modified object. Optional.
SourceRecId	Applicable to Entity type, this is a primary key of a source table for the entity. Optional. This parameter shall be used in conjunction with EntityType.
EntityType	The entity type of modified entity object (Customer, Vendor, Product).
Code	The code of a modified object. Applicable to Entity type. Optional. This parameter shall be used in conjunction with EntityType and CompanyKey.
CompanyKey	Primary key of the Company record the modified object belongs to. Optional.
TimeStamp	Populated by the Pricing Service, this parameter indicates the Time when the message was received by the Pricing Service.

Message API

RMx Pricing service supports two different strategies of updating data in its cache. The second strategy comprises methods for receiving and processing modified data in serialized form, thus eliminating the need to ever contact database directly. A running instance of RMx Pricing cannot mix the strategies: it can be configured to use only one of them.

Add Data Message Method

Namespace: RMxPricing

Syntax

```
DataMessage AddDataMessage(DataMessage message, string clientDataSource);
```

Parameters

Name	Type	Description
message	DataMessage	Contains a modified object in serialized form.
clientDataSource	string	Combination of domain, SQL server name and DB name to uniquely identify a data source

ReturnValue

A DataMessage object containing the processed message status and possibly an error message.

Usage

```
public async Task OneDataMessage()
{
var dataSource = "dataSource";
var requestBody1 = new DataMessage
{
        CacheObjectType = CacheObjectType.TradeAgreementVersion,
       ChangeType = PersistType.Update,
        SyncProgress = SyncProgress.SyncPart,
       CompanyCode = "USMF",
        MessageStatus = MessageStatus.Created,
       MessageId = 1,
        SerializedObject = "[{\"MemberKey\":5637161827,\"AgreementKey\":5637155828,\"UnitOfMeasureKey\":null," +
"\"StartDate\":null,\"EndDate\":null,\"Type\":1,\"IncludeAll\":0,\"CompanyKey\":5637145326}]"
};


DataMessage response = await _rmxPricingClient.AddDataMessageAsync(dataSource, requestBody1);
}
```

Postman

Http method	POST
URL	/api/pricingEngine/message/addDataMessage
Query parameters	clientDataSource=

Request Body


```
{
"messageId": 1,
"identifier": "00000000-0000-0000-0000-000000000000",
"messageStatus": 0,
"cacheObjectType": 9,
"syncProgress": 0,
"changeType": 1,
"primaryKeys": [
5637155828
   ],
"companyCode": "usmf",
"serializedObject": "[{\"TradeAgreementVersionKey\":5637155828,\"IsManual\":0,\"OrderUsageLimit\":null,

\"Code\":\"TA-000000014\",\"TradeAgreementType\":3,\"Currency\":\"USD\",\"UnitOfMeasureKey\":null,
\"PriorityKey\":null,\"StartDate\":\"2021-07-05 00:00:00\",\"EndDate\":\"2021-07-31 23:59:59\",
\"InvalidForPricing\":0,\"CompanyKey\":5637145326,\"AgreementTag\":\"\"}]",
"error": null,
"messageStatusesLB": ""
}
```


Response Status	200

Response Body


```
{
"messageId": 1,
"identifier": "00000000-0000-0000-0000-000000000000",
"messageStatus": 1,
"cacheObjectType": 9,
"syncProgress": 0,
"changeType": 1,
"primaryKeys": [
5637155828
   ],
"companyCode": "usmf",
"serializedObject": null,
"error": null,
"messageStatusesLB": ""
}
```


Add Data Messages Method

Namespace: RMxPricing

Syntax

```
List<DataMessage> AddDataMessages(List<DataMessage> messages, string clientDataSource);
```

Parameters

Name	Type	Description
messages	List<DataMessage>	Collection of DataMessages, each of which contains a modified object in serialized form.
clientDataSource	string	Combination of domain, SQL server name and DB name to uniquely identify a data source

ReturnValue

A collection of DataMessage objects, each of which contains the processed message status and possibly an error message.

Usage

```
public async Task MultiDataMessage()
{
var dataSource = "dataSource";
var requestBody1 = new DataMessage
{
        CacheObjectType = CacheObjectType.TradeAgreementMemberInclusion,
       ChangeType = PersistType.Update,
        SyncProgress = SyncProgress.SyncPart,
       CompanyCode = "USMF",
        MessageStatus = MessageStatus.Created,
       MessageId = 1,
        SerializedObject = "[{\"MemberKey\":5637161827,\"AgreementKey\":5637155828,\"UnitOfMeasureKey\":null," +
"\"StartDate\":null,\"EndDate\":null,\"Type\":1,\"IncludeAll\":0,\"CompanyKey\":5637145326}]"
};


var requestBody2 = new DataMessage
{
        CacheObjectType = CacheObjectType.TradeAgreementVersion,
       ChangeType = PersistType.Update,
        SyncProgress = SyncProgress.SyncPart,
       CompanyCode = "USMF",
        MessageStatus = MessageStatus.Created,
       MessageId = 2,
        SerializedObject = "[{\"TradeAgreementVersionKey\":5637155828,\"IsManual\":0,\"OrderUsageLimit\":null,\"Code\":\"TA-000000014\"," +
"\"TradeAgreementType\":3,\"Currency\":\"USD\",\"UnitOfMeasureKey\":null,\"PriorityKey\":null," +
"\"StartDate\":\"2021-07-05 00:00:00\",\"EndDate\":\"2021-07-31 23:59:59\",\"InvalidForPricing\":0," +
"\"CompanyKey\":5637145326,\"AgreementTag\":\"\"}]"
};


var requestList = new List<DataMessage> {requestBody1, requestBody2};


IList<DataMessage> response = await _rmxPricingClient.AddDataMessagesAsync(dataSource, requestList);
}
```

Postman

Http method	POST

URL	/api/pricingEngine/message/addDataMessages
Query parameters	clientDataSource=

Request Body	[
	{
	"messageId": 1,
	"identifier": "00000000-0000-0000-0000-000000000000",
	"messageStatus": 0,
	"cacheObjectType": 9,
	"syncProgress": 0,
	"changeType": 1,
	"primaryKeys": [
	5637155828
	],
	"companyCode": "usmf",
	"serializedObject": "[{\"TradeAgreementVersionKey\":5637155828,\"IsManual\":0,\"OrderUsageLimit\":null,
	\"Code\":\"TA-000000014\",\"TradeAgreementType\":3,\"Currency\":\"USD\",\"UnitOfMeasureKey\":null,
	\"PriorityKey\":null,\"StartDate\":\"2021-07-05 00:00:00\",\"EndDate\":\"2021-07-31 23:59:59\",
	\"InvalidForPricing\":0,\"CompanyKey\":5637145326,\"AgreementTag\":\"\"}]",
	"error": **null**,
	"messageStatusesLB": ""
	},
	{
	"messageId": 2,
	"identifier": "00000000-0000-0000-0000-000000000000",
	"messageStatus": 0,
	"cacheObjectType": 11,
	"syncProgress": 0,
	"changeType": 1,
	"primaryKeys": [
	5637161827
	],
	"companyCode": "usmf",
	"serializedObject": "[{\"MemberKey\":5637161827,\"AgreementKey\":5637155828,\"UnitOfMeasureKey\":null,
	
	\"StartDate\":null,\"EndDate\":null,\"Type\":1,\"IncludeAll\":0,\"CompanyKey\":5637145326}]",
	"error": **null**,
	"messageStatusesLB": ""
	},
	{
	"messageId": 3,
	"identifier": "00000000-0000-0000-0000-000000000000",
	"messageStatus": 0,
	"cacheObjectType": 13,
	"syncProgress": 0,
	"changeType": 2,
	"primaryKeys": [
	5637149828
	],
	"companyCode": "usmf",
	"serializedObject": "",
	"error": **null**,
	"messageStatusesLB": ""
	}
	]

Response Status	200

Response Body


```
[
   {
"messageId": 1,
"identifier": "00000000-0000-0000-0000-000000000000",
"messageStatus": 1,
"cacheObjectType": 9,
"syncProgress": 0,
"changeType": 1,
"primaryKeys": [
5637155828
      ],
"companyCode": "usmf",
"serializedObject": null,
"error": null,
"messageStatusesLB": ""
   },
   {
"messageId": 2,
"identifier": "00000000-0000-0000-0000-000000000000",
"messageStatus": 1,
"cacheObjectType": 11,
"syncProgress": 0,
"changeType": 1,
"primaryKeys": [
5637161827
      ],
"companyCode": "usmf",
"serializedObject": null,
"error": null,
"messageStatusesLB": ""
   },
   {
"messageId": 3,
"identifier": "00000000-0000-0000-0000-000000000000",
"messageStatus": 1,
"cacheObjectType": 13,
"syncProgress": 0,
"changeType": 2,
"primaryKeys": [
5637149828
      ],
"companyCode": "usmf",
"serializedObject": null,
"error": null,
"messageStatusesLB": ""
   }
]
```


Data Message Class

Syntax

```
public class DataMessage
{
  public long MessageId{ get; set; }
  public Guid Identifier { get; set; }
  public MessageStatus MessageStatus { get; set; }
  public CacheObjectType CacheObjectType { get; set; }
  public SyncProgress SyncProgress { get; set; }
  public PersistType ChangeType{ get; set; }
  public List<long> PrimaryKeys { get; set; }
  public string CompanyCode { get; set; }
  public string SerializedObject { get; set; }
  public string Error { get; set; }
}
```

Properties

Name	Type
MessageId	A parameter identifying the message by an integer number
Identifier	A parameter identifying the message by a GUID
MessageStatus	Processing status of the message
CacheObjectType	A parameter identifying the cache object type.
SyncProgress	Indicates if the current message represents part of the overall synchronization, partial update or it ends the synchronization process
ChangeType	The enumeration denoting the type of object modification (Created, Updated, Deleted).
PrimaryKeys	The primary keys of a modified objects. Optional.
CompanyCode	The company code the modified object belongs to.
SerializedObject	A string representing either single or a collection of modified objects in serialized form.
Error	Error message generated in course of processing the data message.

Accrual API

RMx Pricing service exposes functions to initialize/Complete/Cancel of an accrual run. It also provides a function necessary to process Stepped Breaks in accrual calculations.

Initialize Accrual Run Method

Namespace: RMxPricing

Syntax

```
string InitializeAccrualRun(string clientDataSource, InitializeAccrualRunRequest body);
```

Parameters

Name	Type	Description
body	**InitializeAccrualRunRequest**	**Class includes CompanyCode, CurrencyCode, RunType, RunLevel.**
clientDataSource	string	Combination of domain, SQL server name and DB name to uniquely identify a data source

ReturnValue

A string object with an error message if any errors were detected.

Usage

```
public async Task InitializeAccrualRun()
{
var dataSource = "dataSource";
var request = new InitializeAccrualRunRequest
{
      CompanyCode = "USMF",
       CurrencyCode = "USD",
      RunLevel = RunLevel.Incremental,
      RunType = ""
};
   string response = await _rmxPricingClient.InitializeAccrualRunAsync(dataSource, request);
}
```

Postman

Http method	POST
URL	/api/pricingEngine/accrual/initializeAccrualRun
Query parameters	clientDataSource=
Request Body	{ "companyCode": "USMF", "currencyCode": "USD", "runLevel": "Incremental", "runType": "" }
Response Status	200
Response Body	""

Complete Accrual Run Method

Namespace: RMxPricing

Syntax

```
public Dictionary<long, BreakQuantities> CompleteAccrualRun(string clientDataSource, StandardRequest body);
```

Parameters

Name	Type	Description
body	StandardRequest	Class includes CompanyCode, CurrencyCode, RunType.
clientDataSource	string	Combination of domain, SQL server name and DB name to uniquely identify a data source

Return Value

A dictionary object with BreakQuantity for stepped break components used in the accrual run. They have to be persisted in the RMx database to be used in the subsequent accrual runs.

Usage

```
public async Task CompleteAccrualRun()
{
var dataSource = "dataSource";
var request = new StandardRequest
{
      CompanyCode = "USMF",
       CurrencyCode = "USD",
      RunType = ""
};


IDictionary<string, BreakQuantities> response = await _rmxPricingClient.CompleteAccrualRunAsync(dataSource, request);
}
```

Postman

Http method	POST
URL	/api/pricingEngine/accrual/completeAccrualRun
Query parameters	clientDataSource=
Request Body	{ "companyCode": "USMF", "currencyCode": "USD", "runType": "" }
Response Status	200
Response Body	{ "1234": { "qtyOverall": 100.0, "qtyTriggers": { "111": 10, "222": 15, "333": 75 } } }

Cancel Accrual Run Method

Namespace: RMxPricing

Syntax

```
public string CancelAccrualRunAsync(string clientDataSource, StandardRequest body);
```

Parameters

body	**StandardRequest**	**Class includes CompanyCode, CurrencyCode, RunType.**
clientDataSource	string	Combination of domain, SQL server name and DB name to uniquely identify a data source

ReturnValue

A string object with any detected errors, or an empty string.

Usage

```
public async Task CancelAccrualRun()
{
var dataSource = "dataSource";
var request = new StandardRequest
{
      CompanyCode = "USMF",
       CurrencyCode = "USD",
      RunType = ""
};


string response = await _rmxPricingClient.CancelAccrualRunAsync(dataSource, request);
}
```

Postman

Http method	POST
URL	/api/pricingEngine/accrual/cancelAccrualRun
Query parameters	clientDataSource=
Request Body	{ "companyCode": "USMF", "currencyCode": "USD", "runType": "" }
Response Status	200
Response Body	""

Break Quantities Class

Syntax

```
public class BreakQuantities
{
public decimal QtyOverall{ get; set; }
public Dictionary<long, decimal> QtyTriggers { get; set; }
}
```

Properties

Name	Type
QtyOverall	Total contribution of all the priced document lines in accrual run to the Overall break quantity, which matched the trade agreement price for the current price break.
QtyTriggers	Collection of total contributions of all the accrual priced lines to break trigger quantities. The index of the dictionary corresponds to MemberEntity keys for corresponding break triggers.

Standard Request Class

Syntax

```
public class StandardRequest
{
public string CompanyCode { get; set; }


public string CurrencyCode { get; set; }


public string RunType { get; set; }
}
```

Properties

Name	Type
CompanyCode	Company code for the accrual run
CurrencyCode	Currency code
RunType	>Code of a RunType for the accrual run

Initialize Accrual Run Request Class

Syntax

```
public class InitializeAccrualRunRequest
{
public string CompanyCode { get; set; }


public string CurrencyCode { get; set; }


public string RunType { get; set; }
public RunLevel RunLevel { get; set; }
}
```

Properties

Name	Type
CompanyCode	Company code for the accrual run
CurrencyCode	Currency code
RunType	Code of a RunType for the accrual run
RunLevel	RunLevel enumeration (Incremental, Full accrual run)