Data Monitor SDK

Getting Started

Installation

To consume the es6 module dist, run the following installation commands:

yarn: `yarn add data-monitor-javascript-sdk`

npm: `npm install data-monitor-javascript-sdk`

Usage

Download the evergreen content of dataMonitor through the SDK entrypoint, `init` , then subscribe to data.

```
import {init, on} from 'data-monitor-javascript-sdk';

// initialize dataMonitor as early as possible within app's lifecycle to get the
best performance
await init({idpid});

async function subscribe(){
  // safety net, or if you don't want data monitor to be initialized globally.
  await init({idpid});

  // start data subscriptions
  const subscriptionCollection = on({
    symbols: ['FDS-US', 'QQQ-USA'],
    fields: ['LAST_PRICE', 'DISPLAY_CURRENCY', 'PERCENT_CHANGE'],
    success: (message, subscription) => {
        const {LAST_PRICE, PERCENT_CHANGE} = message.data;
        console.log(`The price of ${subscription.symbol} is now ${LAST_PRICE} (${
{PERCENT_CHANGE}), a percentChange of ${PERCENT_CHANGE}`.);
    },
  });
}
```

In this simplistic example, we subscribe to two "symbols", 'FDS-US' (FactSet Research Systems Inc.) and 'QQQ-USA' (Invesco QQQ Trust), and ask for three fields (data points), 'LAST_PRICE' (current stock price), 'DISPLAY_CURRENCY' (the output currency of any pricing data),

'PERCENT_CHANGE' (The percentage change between current price and the previous close price).

When data monitor receives available information about either symbol, it will invoke the `success` callback with an object containing all of the current field information for the symbol that it is referencing.

Note that the order of data received is not guaranteed. The `success` function may be executed multiple times for the same symbol, depending on the rate of realtime data change. Each time `success` is invoked, not all requested fields may yet be present.

Authentication

FactSet supports Single Sign On (SSO) via the industry standard framework, SAML 2.0. SAML 2.0 is an Open Framework Standard for authentication between two domains via SAML-based protocols and security tokens. Assertions are passed between the principal, typically the end user, their Identity Provider (Your Domain) and Service Provider (FactSet). SAML2.0 allows for web-based authentication, which reduces administration of multiple accounts, passwords and sign-on to external entities.

Initialization

Environment Variables

If consuming the dataMonitor sdk through a modern build system that uses a bundler like webpack, rollup, or parcel, you may leverage the following environment variables:

`DATA_MONITOR_ENV`

This variable controls the execution environment of the SDK. Defaults to `PROD`.

Valid values for `DATA_MONITOR_ENV` are:

- `STAGING`
- `PROD`

API

`init(initOptions)`

`initOptions`

- `idpid`: `string`
  - provide the idpid of your organization in order to authenticate the interactions with FactSet services
- `environment`: `Environment`
  - controls the execution evironment of the SDK
  - not necessary if using the `DATA_MONITOR_ENV` environment variable at build time
  - takes the same values as the `DATA_MONITOR_ENV` environment variable.
    - an enum, `Environment`, containing the valid values is exposed by the sdk for convenience.

    - if not provided and the environment variable is not used, will default to `PROD`

- `globalOptions`: `SubscriptionOptions`
    - any options that are valid to provide to a subcription function (`on` / `once`) can be provided here. If provided, these will become the global default options for all subscriptions created. The global options can be overriden by explicitly passing an option to the given subscription function.
    - refer to the list of options in the Subscription Entrypoint Documentation for valid parameters
    - **Note: Use this functionality carefully. Registering global subscription options will impact other components within an application. This is generally something that should be done by an application and not an individual component.**

Best Practice

When the init function is called the first time, Data Monitor will authenticate the user (read Authentication section for more information). It's recommended to setup your init call differently if your app requries:

- **Performance**. If the data request needs to be as fast as possible than authenticating early in the app life-cycle will help.
- **Consistent Settings**. If the client is setting global subscription options that all subscriptions must share, doing it early in one central spot would be helpful.
- **Lazy Loading of Data Monitor**. For a single page app where data monitor might not be used right away or only after the user navigates to another section of the app could be a waste of resources doing that too early on.

Example

```
import {init, Environment} from 'data-monitor-javascript-sdk';

await init({
  // provide the idpid for authentication
  idpid,
```

```
  // set the runtime environment to the staging environment
  environment: Environment.STG,

  // set a default subscription currency option to the JPY currency
  globalOptions: {
    currency: 'JPY',
  },
});
```

DataMonitor Subscription Entrypoints

```
// Provides realtime updating data. Subscriptions will continue to emit updating
data until canceled.
dataMonitor.on;

// Provides one time snapshot data. Subscriptions will not emit updates after
firing the `complete` callback.
dataMonitor.once;
```

`on({symbols, fields, success, error, notify, options}): SubscriptionCollection`

Subscribes to data based on the symbols and fields provided and will fire the supplied callback each time data is retrieved or updated.

`once({symbols, fields, success, error, complete, notify, options}): SubscriptionCollection`

Retrieves snapshot data based on the symbols and fields provided and will fire the supplied callback once the data is retrieved.

- `symbols` - a string or an array of strings containing the various symbols to subscribe to, i.e. `FDS-USA` or `['FDS-USA', 'AAPL-USA']`

  Please see note on input symbol casing preservation.

- `fields` - an array where each element contains either a string id for a data point or an object containing the id of a data point as well as any other meta data to override

- `success` - a function to be called when data is received for each symbol
  - `message` - an object containing a data object with all the data points which have been recieved at the point your callback was fired

    - `subscription` - the `Subscription` tied to the emitted message
- `error` - a function to be called when an error is thrown
    - `message` - an object containing any details regarding the error thrown
    - `subscription` - the `Subscription` tied to the emitted error
- `notify` - a function to be called when a dataService emits a non subscription specific message
    - `source` - an identifier for the dataService that emitted the message
    - `message` - the message emitted by the dataService
- `complete` - a function which is called only after all the requests have completed
    - `errors` - an object map containing an array of errors that occurred by corresponding symbol
    - `messages` - an object map containing the requsted data by corresponding symbol
    - `subscriptions` - an object map of `Subscription`s by corresponding symbol

  Please see note on input symbol casing preservation.
- `options` - an object containing various settings that may augment the call to `on` / `once`
    - `currency` (string) - an ISO code to represent the desired output currency for any currency fields requested, i.e. `'USD'` or `'EUR'`
        - The currency can also be set to `'LOCAL'` to leave the currency in the exchange local or `'NONE'` to disable currency conversion
        - Defaults to `'LOCAL'`
    - `timezone` (string) - an Olson entry to represent the output timezone for all date and time fields
        - The timezone can also be set to `'LOCAL'` to default to the user's machine timezone or to `'NONE'` to disable timezone conversion
        - Defaults to `'LOCAL'`
    - `skipExtend` (boolean) - dictates if messages from backends should be emitted as is without being extending onto the response of other backends or previous data ticks.

- `timeout` (number | boolean) - time (in ms) for a request to DM to timeout.
  - Defaults to never. If `true`, will default to 28 seconds.
  - When a timeout occurs, the `complete` callback will be fired with any data and errors dataMonitor has collected up until that point. Utilize `dataMonitor.isTimeout(errors)` to determine if a timeout has occured.
- `britishCurrencyInPence` (boolean) - when true and the underlying symbol is british and is listed as pence on its source exchange, show pricing data in pence instead of GBP.
  - Defaults to `false`
- `previousCloseAdjusted` (boolean) - when true, show the closing price for a stock's value after accounting for any corporate actions.
  - Defaults to `false`
- `usePreviousClose` (boolean) - when true, show the previous day's close price (2 days ago) during the interim period between active market hours.
  - Defaults to `false`
- `fixedIncomePricesInDecimal` (boolean) - when true and the underlying symbol is a fixed income issue type, normalize pricing values from the fractional representation to a decimal value.
  - Defaults to `false`
- `autoPause` (boolean) - determines whether the subscription should automatically pause and resume when Data Monitor detects a change in page visibility Defaults to `true`

Input Symbol Casing Preservation

Although the returned subscriptions collection will preserve the number and casing of a users input symbols, any subscriptions with identical symbols (e.g. fds-us, FDS-us) will share the same underlying data.

Internally, all symbols are normalized to upper-case. This means that the `complete` response map will contain one entry per unique symbol and the keys will all be upper-case.

Subscription

A `subscription` is an object that represents a subscription in Data Monitor for a specific symbol. It contains meta data about the request as well as methods to manage the request.

Best Practices

It is vital that unused subscriptions are cleaned up appropriately to maintain the best application performance and to ensure efficient use of network resources.

Be sure to:

- call init function to kick off authentication early in your application's life cycle if performance of data requests is necessary.
- hook into your framework's lifecycle hook to cancel your subscription (e.g. Vue's `destroyed`, angular's `$onDestroy`) when a component is cleaned up.
- implement logic in large grids to only keep subscriptions open for visible rows.

API

`cancel()`

Will close any open real-time subscriptions, cancel any pending http requests, and remove the subscription and it's data from memory.

`update(options)`

Takes an object of the following:

- `currency` - an ISO code to represent the desired output currency for any currency fields requested, i.e. `'USD'` or `'EUR'`
    - The currency can also be set to `'LOCAL'` to leave the currency in the exchange local or `'NONE'` to disable currency conversion

- `timezone` - an Olson entry to represent the output timezone for all date and time fields
  - The timezone can also be set to `'LOCAL'` to default to the user's machine timezone or to `'NONE'` to disable timezone conversion
- `fields` - an array of field ids or definitions to set onto the subscription. This **will not** cause data to refetch, but will prevent re-emitting static data on subsequent ticks if that field id is not present in the new field set.

`pause()`

Used to temporarily pause streaming data. If there are no other active listeners, `pause` will seamlessly close the real-time connection until the subscription is resumed.

`resume()`

Used to resume streaming data on a paused subscription. `Resume` will reattach listeners and seamlessly restore the real-time connection if appropriate. Resume will not refetch static fields.

SubscriptionCollection

A SubscriptionCollection instance is returned from dataMonitor.on and dataMonitor.once, and is a keyed object of subscription symbols to `Subscription` objects containing convenience functions that apply `update`, `cancel`, `pause`, and `resume` to all member subscriptions.

Examples

```
const symbols = ['FDS-USA', 'AAPL-USA', 'VOD-LON'];
const fields = ['COMPANY_NAME', 'LAST_PRICE', 'LAST_TIME', 'LAST_DATE'];

// subscribe to data
const subscriptionCollection = dataMonitor.on({
  symbols,
  fields,
  options: {
    // return british company pricing data in pence instead of GBP
    britishCurrencyInPence: true,
    // return pricing data in the local currency of the source exchange (this is
the default if omitted)
```

```
    currency: 'LOCAL',
    // return time/date fields matching the user's machine settings (this is the
default if omitted)
    timezone: 'LOCAL'
  },
  success: (message, subscription) => {
    console.log(message, subscription);
  }
});

// update all subscription date/time fields to UTC timezone
subscriptionCollection.update({timezone: 'UTC'});

const appleSubscription = subscriptionCollection['AAPL-USA'];
// update the apple subscription's pricing data to JPY currency
appleSubscription.update({currency: 'JPY'});

// pause/resume realtime data updates based on page visibility
// This is now handled automatically by Data Monitor with autoPause flag on,
leaving the context here for reference.
// document.addEventListener('visibilitychange', isVisible => {
//   if (isVisible) {
//     subscriptionCollection.resume();
//   } else {
//     subscriptionCollection.pause();
//   }
// });

// cancel the realtime subscriptions after a period of time
setTimeout(() => subscriptionCollection.cancel(), 30000);
```

Data Point

A data point (field) describes a piece of data (example: `price`), and are registered to Data Monitor through the `dataPoint` service.

To subscribe to a field in Data Monitor, check out the `Subscription` section of the docs; this document refers to usage of the `dataPoint` service, specifically.

Getting Started

```
import {init, dataPoint} from 'data-monitor-javascript-sdk';

init().then(async () => {
  const priceField = await dataPoint.get('LAST_PRICE');
  console.log(priceField);
};
```

API

`register(field)`

Register a data point or collection of data points

- `field` - an object representing a data point or an array of objects

For more information refer to the 'Creating or Modifying Data Points' section of the docs below.

`get(item)`

Gets the requested data point.

- `item` - a string representing the data point's id, a partial data point object that contains a data point id to extend, or an array of strings and/or objects

Returns

Returns a `Promise` which resolves with the requested data point(s). If overrides were provided in the form of an object they would be extended onto the corresponding data points.

Examples

```
import {dataPoint} from 'data-monitor-javascript-sdk';

async function registerField() {
  const annualDivField = await dataPoint.get('ANNUAL_DIV');

  dataPoint.register(Object.assign(annualDivField, {
    endpoint: 'fql',
    endpointProperty: 'MY_FIELD_PROP_OVERRIDE',
  }));
}
```

Creating or Modifying Data Points

To see a list of off-the-shelf data points (fields) that can be used, see the 'Data Monitor Fields' section.

Data Point Definition

The data point definition has several properties that can be defined. The information that follows describes all of the properties of a data point definition.

id - `string` - required

A unique, preferably camel-cased, word to identify this data point.

There is no default value for this property.

Example:

```
{
  id: 'PREVIOUS_CLOSE_PRICE';
}
```

endpoint - `string` - required

The data service used to fetch this data point. Valid values are as follows:

`fql` - This data service uses the FQL service to fetch data. If needing to fetch an attribute, the best practice is to use the `GET_ATTRIBUTE_VALUE` FQL function. If needing multiple attributes from the same FQL function, the main function can be cached as a variable and the `GET_ATTRIBUTE_VALUE` calls can reference the variable instead.

type - `string` - optional

The type of data returned by this data point. The chosen type is used to determine how to parse, normalize, and format the data that is processed by Data Monitor.

The valid values are `string`, `number`, `date`, `time`, or `blob`

The default value is `'string'`

Type `date` fields return strings in the format of `YYYYMMDD`.

Type `time` fields return strings in the format `HHMMSSmmm`.

fxRate - `string` - optional

The corresponding exchange rate to use for this dataPoint. In order to accurately currency convert historical values, they must use historical exchange rates. DataMonitor currently supports the following exchange rates:

```
enum FXRates {
  NOW = 'NOW',
  YESTERDAY = 'YESTERDAY',
  YESTERDAY_PREV_CLOSE = 'YESTERDAY_PREV_CLOSE',
  ONE_WEEK_AGO = 'ONE_WEEK_AGO',
  ONE_MONTH_AGO = 'ONE_MONTH_AGO',
  THREE_MONTHS_AGO = 'THREE_MONTHS_AGO',
  SIX_MONTHS_AGO = 'SIX_MONTHS_AGO',
  ONE_YEAR_AGO = 'ONE_YEAR_AGO',
  YEAR_END = 'YEAR_END',
  WEEK_END = 'WEEK_END',
  MONTH_END = 'MONTH_END',
  QUARTER_END = 'QUARTER_END',
}
```

The above enum is exported from dataMonitor so in-app defined dataPoints can leverage it.

timezoneConvertible - boolean - optional

If specified as `true`, the field will be run through the timezone conversion logic as long as `options.timezone` is being used and is not `NONE`.

name - `string` - optional

A display name for this data point.

The default value is a normalized, pascal-cased with spaces, version of the `id` property

Example: `'Previous Close Price'`

cacheable - `boolean | number(in ms) | function()` - optional

Determine whether the field can be cached through a data store for a given period of time. When the time `DataMonitor.on` or `DataMonitor.once` has been called, they will first check if data is available and not expired from data store instead of fetching data again from data services repeatedly. This will reduce redundant service requests.

Store will only be enabled if either `options.useStore` is set to `true`, or a feature flag of `useStore` is passed in.

If you are implementing your own data services, read this.

Example:

```
//data point definition
{
  id: 'cacheableField',
  ...
  cacheable: true, //field data will be cached permanently.
}

{
  id: 'cacheableFieldWithExpiration',
  ...
  cacheable: 600000, //field data will be stored for 600000ms (10 minutes)
}
```

computedFunction - `function` - optional

A function to be called that returns this data point's data. This is used in conjunction with dependencies in order to fetch data and then perform a calculation to return a new value. A good example is the `change` data point that subtracts the current price from the previous close price. The input to the computed function is an object containing all the data Data Monitor has collected so far for that symbol. It's possible that the data needed for the computation may not have been collected yet and it's a developers responsibility to add necessary protections in any computed function calculations.

A computed function will not run unless data for all of its required `dependencies` are present.

There is no default value for this property.

Example:

```
// Computed function definition
function change({LAST_PRICE, PREVIOUS_CLOSE_PRICE}) {
  return LAST_PRICE - PREVIOUS_CLOSE_PRICE;
}
// Data point definition
{
  id: 'CHANGE',
  computedFunction: change,
  dependencies: ['LAST_PRICE', 'PREVIOUS_CLOSE_PRICE']
}
```

dependencies - `array` (string | object) - optional

An array of other data point IDs that need to be fetched along with this data point. Data Monitor will make sure to fetch these data points if they were not explicitly requested. Dependencies are typically used with computed data points and certain data types. Dependencies are important for `'date'`, and `'time'` data points.

For more information about these types, see the `Date & Time Data Points` sections below. FQL Attribute data points also leverage dependencies. More information about FQL Attributes can be found in the `FQL Attribute Data Points` section below.

The `dependencies` array can be just the singular id or an object containing the `id`. The object form may specify an `optional` property, which when used with a `computedFunction` dictates if the function should be able to run when the dependency's value is not present. By default, all dependencies are treated as required unless explicitly marked as optional.

The default value is `[]`.

Example:

```
{
  id: 'MY_COMPUTED_FIELD',
  computedFunction: function(data) {
    let calculation = data.REQUIRED_DEPENDENCY * 25;
    if (data.OPTIONAL_DEPENDENCY) {
      calculation += data.OPTIONAL_DEPENDENCY;
    }
```

```
    return calculation;
  },
  dependencies: ['REQUIRED_DEPENDENCY', {id: 'OPTIONAL_DEPENDENCY', optional:
true}],
}
```

Endpoint Structure

The dataPoint definition contains a object property `endpoint`, which itself can have the properties of `default` and `fallback`.

`endpoint.default` - The default service for the field. If there is no fallback, there is no need to have `endpoint` be an object.

`endpoint.fallback` - In some cases you want to try a service first without a guarantee of data because of net performance gain. `endpoint.fallback` is meant to cover the case where the original target service did not return appropriate data. `fallback` follows the same structure as other metadata grouped under the `endpoint` object,

In the following example, the `ANNUAL_DIV` field has no fallback and therefore `endpoint` is a string. For the `NAV_DATE` field, DataMonitor will request `ETP_NAV().DATES` from `fql` first. Only if this first choice fails to return appropriate data, dataMonitor will fetch `MSTAR_PRICE_DATE_MR` from `fql` in as a backup.

```
{
  id: 'ANNUAL_DIV',
  name: 'Ann Div',
  type: 'currency',
  endpoint: 'fql',
  endpointProperty: 'FWC_DIVS_PS()',
},
{
  id: 'NAV_DATE',
  name: 'NAV Date',
  type: 'date',
  endpoint: {
    default: {
      endpoint: 'fql',
      endpointProperty: 'ETP_NAV().DATES',
    },
    fallback: {
      endpoint: 'fql',
      endpointProperty: 'MSTAR_PRICE_DATE_MR("NAV")\
```

```
        @FDI_UNIT_PRICE_DATE_MR("FEXP","NAV_DATE","YYYYMMDD")\
        @P_NAV(NOW).DATES',
    },
  },
},
```

endpointProperty - `string` **- optional/required**

This property contains the field which contains the data that was returned by the data service for this data point. This can be either the FQL code for the FQL service, for example, or it could be a JSON property from a JSON based service. This value should be case-sensitive so that Data Monitor can retrieve the correct value

This property is optional if `endpointProperty` is defined within the `endpoint` object. It is required if `endpoint` is a `string`.

This property has no default value.

Example: `'LAST_1'` or `'FWC_SEC_NAME()'`

endpointReplacements - `object` **- optional**

An object of key/value pairs where the key is a string found in the `endpointProperty` and the value is what Data Monitor should replace that string with.

`endpointReplacements` is always a root level property, and cannot be cascaded into the dynamic endpoint routing.

This is typically used for FQL functions to allow applications to reuse a data point definition but pass it new parameters for the FQL function with each subsequent usage

This property has no default value.

Example:

Given an endpointProperty of `'EXRATE(fromCurrency,toCurrency,NOW)'`. The `endpointReplacements` property could be as follows: `{ 'fromCurrency': 'LOC', 'toCurrency': 'LOC' }`. This would allow applications to override these replacements with user selected currencies while keeping the default for both of the FQL parameters as `'LOC'` in case the application does not provide a particular override. The following code shows how an application could pass new replacements for a data point.

```
// Modify the endpoint replacements when subscribing
// Provide a unique field for the data to return in if overriding the same data
point multiple times
dataMonitor.on({
  symbols: ['FDS-USA'],
  fields: [
    {
      id:'EXCHANGE_RATE_NOW',
      endpointReplacements: {
        fromCurrency: 'USD',
        toCurrency: 'EUR'
      }
    }
  ],
  ...
});
```

Currency Data Points

Data points that support currency conversion are represented by the `FXRate` property of the definition.

Example:

```
{
  id: 'PRICE_ONE_WEEK_AGO',
  type: 'number',
  fxRate: FXRates.ONE_WEEK_AGO,
  endpoint: 'fql',
  endpointProperty: 'FWC_PRICE_1_WEEK_AGO()',
},
{
  id: 'PRICE_ONE_YEAR_AGO',
  type: 'number',
  fxRate: FXRates.ONE_YEAR_AGO,
  endpoint: 'fql',
  endpointProperty: 'FWC_PRICE_YEAR_AGO',
}
```

Date & Time Data Points

Date data points are represented by the `type: 'date'` property of the definition while Time data points have `type: 'time'` . In order
for Data Monitor to be able to perform the proper time zone conversions on the data, the date data points need to have a dependency on the corresponding time data point in order for cross day conversions to be accurate.

Similar to currency data points, Data Monitor needs to know some information about the incoming data in order to know how to properly convert it. Data Monitor will know to fetch that additional data for timezone conversions by specifying the `timezoneConvertible` field property as `true` .

Example:

```
{
    id: 'MY_DATE',
    type: 'date',
    timezoneConvertible: true,
    dependencies: [ 'MY_TIME' ],
    endpoint: 'fql',
    endpointProperty: 'MY_FQL_DATE_FORMULA'
}
{
    id: 'MY_TIME',
    type: 'time',
    timezoneConvertible: true,
    endpoint: 'fql',
    endpointProperty: 'MY_FQL_TIME_FORMULA'
}
```

FQL Attribute Data Points

In FQL, a common practice is to return data through attributes of the requested FQL code. Data Monitor uses the `services/Fql` Lima service which does not allow all attributes to be returned by accessing `attributes=*` in the request. A way to access attributes from FQL is by using the `GET_ATTRIBUTE_VALUE` FQL code. This code takes two parameters, the first is the source FQL code and the second is the attribute name to return the data for.

Example:

```
endpointProperty: 'GET_ATTRIBUTE_VALUE(HTML_CDS_FI_APP_SPREAD_MAIN("curr",""),
\'full_name\')';
```

For improved performance when accessing multiple attributes from a single FQL code, the best practice is to cache the code in a variable as a shared data point and depend on the shared data point for the GET_ATTRIBUTE_VALUE based data points. Notice the `CDS_DATA` variable in the example below.

Example:

```
{
    id: 'CDS_DATA',
    endpoint: 'fql',
    endpointProperty: 'GENERIC_ARRAY CDS_DATA=
HTML_CDS_FI_APP_SPREAD_MAIN("SPREAD_CHG","TENOR")',
    endpointReplacements: { 'SPREAD_CHG': 'CURR', 'TENOR': '' }
},
{
    id: 'CDS_COUNTRY',
    name: 'Country CDS',
    dependencies: [ 'symbol', 'cds_data' ],
    endpoint: 'fql',
    endpointProperty: 'GET_ATTRIBUTE_VALUE(CDS_DATA,\'full_name\')'
}
```

Data Monitor Fields

The following field list represents available fields for subscription:

ID	Type	Supports Conversion	Description
ASK	number		Last ask price. The price a seller is willing to accept for a security. Also known as the offer price. If the security did not quote the field will appear as 0. For US Funds: This field reflects the price at which a MFQS instruments shares can be purchased from the issuer or distributor. The offer price includes the current net asset value per share plus any sales charges.
ASK_CLOSE	number	currency	Last ask price from the last day a given market was open. Methodologies can vary by exchange.
ASK_DATE	date	timezone	Local exchange date of last offer price. Field updates once per day with first offer price.
ASK_EXCH	number		The exchange code of the participating exchange that disseminated the last offer price.
ASK_EXCHANGE_NAME	string		Exchange on which the last ask price was made.
ASK_TIME	time	timezone	

ID	Type	Supports Conversion	Description
			Time in milliseconds and local time of the last offer price.
ASK_VOL	number		Current size number of shares of offer price. For US issues size adjusted by lot size. Other exchanges actual size of quote.
ASK_YIELD	number		Yield received if note is bought at ask price. As provided by the real-time server.
AUTO_TRADE_VWAP	number	currency	Daily cumulative VWAP calculated on all automated trades. Trade price multiplied by shares and divided by cumulative volume on a per trade basis for automatic trades. VWAP is accumulated throughout the trading day and is recalculated as a result of trade corrections or cancellations.
BID	number		Last bid price. An offer made by an investor trader or dealer to buy a security. The bid will stipulate both the price at which the buyer is willing to purchase the security and the quantity to be purchased.
BID_CLOSE	number	currency	Last bid price from the last day a given market was open. Methodologies can vary by exchange.

ID	Type	Supports Conversion	Description
BID_DATE	date	timezone	Local exchange date of last bid price. Field updates once per day with first bid price.
BID_EXCH	number		The exchange code of the participating exchange that disseminated the last bid price.
BID_EXCHANGE_NAME	string		Exchange on which the last bid price was made.
BID_TICK	number		Direction of last bid price.
BID_TIME	time	timezone	Time in milliseconds and in local time of exchange of the last bid price.
BID_VOL	number		Current size number of shares of bid price. For US issues size adjusted by lot size. Other exchanges actual size of quote.
BID_YIELD	number		Yield received if note is bought at the bid price. As provided by the real-time server.
BLOCK_CVOL	number		Cumulative volume of all trades deemed by either exchange rule or market convention as a block trade. Not all exchanges have definitions of block trades. If populated, Block Money Flow fields will also be populated.
BLOCK_TRADE_COUNT	number		Indicates the number of all trades deemed by either

ID	Type	Supports Conversion	Description
			exchange rule or market convention as a block trade. Used in conjunction with Block Trade Volume.
BUYER_DESCRIPTION	string		
BUY_ID	string		Identifier for the broker on the buy side of the trade.
CHANGE	number		The difference between the real-time price and the previous close.
CLOSE_2	number	currency	End price in closing price range. Used for pit session from US futures. Pits have been mostly shutdown and field has limited use.
CLOSE_YIELD	number		Last yield for a bond from the last day a given market was open as provided by the real-time server. Methodologies can vary by exchange.
COMPANY_NAME	string		The company name.
CONTRACT_HIGH	number		Highest price of derivatives contract over life of the contract.
CONTRACT_LOW	number		Lowest price of derivatives contract over life of the contract.
COUNTRY_CODE	string		Country of domicile for a given instrument.

ID	Type	Supports Conversion	Description
COUNTRY_NAME	string		Full name of country e.g. United Kingdom from PROPERCOUNTRY for the country of domicile for a given instrument.
COUPON_RATE	number		Interest rate assigned to a bond when issued.
CURRENCY_ISO	string		Local Currency for a specified security. The international standardized three-letter abbreviation for a country's currency.
CURRENT_YIELD	number		For bonds, this represents the current yield of a bond. Not all exchanges have coverage of the field. For Funds this field is the current yield of UIT consisting of debt securities.
CVOL	number		Cumulative Volume. The accumulated trade volume for all trades eligible ineligible and unofficial on a given trading day. The cumulative volume is expressed in units of one and is the raw number of trades during the trading day.
DAYS_TO_COVER	number		The number of a company's issued shares that are currently sold Short, as expressed by the number of days required to close out all short positions.

ID	Type	Supports Conversion	Description
DAYS_TO_EXPIRATION	number		Number of days to expiration for a derivative.
DAYS_TO_EXPIRATION_DISP	number		Number of days to expiration for a derivative.
DISPLAY_CURRENCY	string		Current output currency of pricing data.
DIVIDEND_YIELD	number		Annual dividend divided by price.
DURATION	number		Effective duration -- lists the price sensitivity to a change in interest rates.
EPS	number	currency	Last twelve months earnings per share.
EPS_FISCAL_QUARTER_ONE	number	currency	Mean EPS estimate for current unreported fiscal quarter.
EPS_FISCAL_YEAR_ONE	number	currency	Mean EPS estimate for current unreported fiscal year.
EXCHANGE	number		
EXCHANGE_MARKET_SIZE	number		Instrument level rule used in conjunction with a multiplier to determine the maximum and minimum order sizes. Can be used in conjunction

ID	Type	Supports Conversion	Description
			with Average Daily Turnover. The Exchange Market Size EMS is set to show the minimum size a market maker must quote in an individual security for all executable and non executable quotes.
EXPECTED_REPORTING_DATE	date		Expected date of the upcoming earnings call as provided by CallStreet.
EXPECTED_REPORTING_TIME	time		Expected time of the upcoming earnings call as provided by CallStreet.
EXPIRATION_DATE	date	timezone	For Derivatives: Expiry Date. For Fixed Income: Maturity Date. The last day that contracts are valid.
EX_DATE_STATUS	string		Indicates the status of a security's corporate action. Potential values: XS =ex-Split; XD = ex-Dividend;. Corporate actions that are realized on the ex-date are capitalized codes. For the London Stock Exchange upcoming corporation actions are indicated by lower case codes starting on the corporate action's record date. On the ex-date the code will be capitalized.
FIFTY_TWO_WEEK_HIGH	number		Highest intraday price for a security within the last 52 weeks.

ID	Type	Supports Conversion	Description
FIFTY_TWO_WEEK_LOW	number		Lowest intraday price for a security within the last 52 weeks.
FINANCIAL_STATUS	number		Indicates the regulatory status of a security trading on US security markets. The code indicates whether a listed stock has submitted its regulatory filings on a timely basis continuing to meet listing standards and/or filed for bankruptcy. See Financial Status table. Field also indicates if LSE security is in a Bid Situation.
GMT_OFFSET	number		Indicates the difference in minutes of the local time of an exchange/source to GMT.
HALT_DESCRIPTION	string		Indicates the reason for the regulatory halts
HALT_INFO	number		Indicates the reason for the regulatory halts
HIGH	number	currency	Indicates the highest intraday price for eligible trades on a given trading day during continuous trading. Prior to market open the field will display zero. If no eligible trades occurred on a given trading day the field will display zero.
HIGH_52WEEK_DATE	date		

ID	Type	Supports Conversion	Description
			Date of the highest intraday price within the last 52 weeks.
HIGH_YIELD	number		Highest intraday yield.
ISO_CODE	string		Three or four character FDS Exchange Code indicating the source of the data. See Exchange Code table for more detail.
ISSUE_DATE	date		The date on which a security is issued.
LAST_DATE	date	timezone	Date in local exchange time of last price update date of last eligible activity. Field updates with first eligible update of the day.
LAST_EXCH	number		The exchange code of the participating exchange that disseminated the last price.
LAST_PRICE	number		Represents the official price of an instrument as derived from exchange trading rules or a representative price for non-exchange traded instruments. This reflects the value of a financial instruments for purposes of portfolio contribution and value
LAST_TICK	number		Indicates the tick trade direction of the last trade. An Uptick occurs when the last price is higher than the

ID	Type	Supports Conversion	Description
			previous price; a Zero up tick is when the last trade price is unchanged from the previous trade price but the prior direction change was up; a Downtick occurs the last price is lower than the previous price; a Zero down tick happens when the last trade price is unchanged from the previous trade price but the prior direction change was down. Note the change indicator only includes eligible trades when determining the tick direction.
LAST_TIME	time	timezone	Time in milliseconds and in local time of exchange of the last price update.
LAST_VOL	number		The number of executed shares of the last trade.
LOW	number	currency	Indicates the lowest intraday price for eligible trades on a given trading day during continuous trading. Prior to market open the field will display zero. If no eligible trades occurred on a given trading day the field will display zero.
LOW_52WEEK_DATE	date		Date of the lowest intraday price within the last 52 weeks.
LOW_YIELD	number		Lowest intraday yield.

ID	Type	Supports Conversion	Description
MARKET_CAPITALIZATION	number	currency	Market cap or market capitalization refers to the total value of all a company's shares of stock.
MARKET_SECTOR	string		Refers to a minor listing division of a market segment. Sector classification according to the London Stock Exchange.
MARKET_SEGMENT	string		Refers to a major listing division of an exchange - where like instruments are grouped - and denotes a unique listing for security identification. Where possible market segment will be identified with the non-operating MIC code. If not possible codes are either drawn from exchange or created by FDS.
MID	number	currency	Average of last bid and offer. Not available for US stocks.
MID_DATE	date	timezone	Local exchange date of last mid price update. Fields update with the first mid price of the day.
MID_HIGH	number	currency	Highest intraday mid price. Field updates in real time.
MID_LOW	number	currency	Lowest intraday mid price. Field updates in real time.
MID_TIME	time	timezone	

ID	Type	Supports Conversion	Description
			Time in milliseconds and local exchange time of mid price update.
MID_YIELD	number		Yield received if note is bought at the mid price. As provided by the real-time server.
MONTH_TO_DATE_PERCENT_CHANGE	number		The percentage change between current price and the price from previous month end's.
MSG_TYPE	string		Indicates the type of message delivered.
NAV	number	currency	The net asset value as provided from the real-time server (where available) else the most recent net asset value from fund providers (Morningstar, Fundata Canada, FE Fundinfo, FactSet Pricing Database, FactSet Funds) based on the user subscription. LOCAL currency equals the Currency ISO of the ticker.
OFFICIAL_ASK_CLOSE	number	currency	The current trading day's last offer price. If an exchange publishes an official closing offer price then the field will populate with the exchange published value. Field is not populated until exchange is closed. Only securities quoting on current trading day will have fields populated. Field populates

ID	Type	Supports Conversion	Description
			once per day at close. During the trading day field reflects a zero value.
OFFICIAL_ASK_CLOSE_VOL	number		Size (number of shares) of quote associated with the last offer price. For US issues size adjusted by lot size. Other exchanges actual size of quote. Field updates once per day at close. During the trading day the field reflects a zero value.
OFFICIAL_BID_CLOSE	number	currency	The current trading day's last bid price. If an exchange publishes an official closing bid price then the field will populate with the exchange published value. Field is not populated until exchange is closed. Only securities quoting on current trading day will have fields populated. Field populates once per day at close. During the trading day field reflects a zero value.
OFFICIAL_BID_CLOSE_VOL	number		Size (number of shares) of quote associated with the last bid price. For US issues size adjusted by lot size. Other exchanges actual size of quote. Field updates once per day at close. During the trading day the field reflects a zero value.
OFFICIAL_CLOSE	number		Indicates the official closing price on a given exchange.

ID	Type	Supports Conversion	Description
			Can be a traded price, an auction price, a calculated price, or any price deemed and distributed by the exchange as the closing price. It only populates for exchanges where the exchange distributes an official price. The field only populates for those instruments that traded on a given day.
OFFICIAL_CLOSE_TIME	time	timezone	Time in milliseconds and in local exchange time of the current trading day's official closing price.
OFF_HOUR_CUMULATIVE_VOLUME	number		After-hours trading cumulative volume.
OFF_HOUR_PRICE	number	currency	After-hours trading price.
OFF_HOUR_SALE_CONDITION	number		Indicates the special off-hours condition of a transaction. Conditions vary by exchange.
OFF_HOUR_VOLUME	number		After-hours trading volume.
ONE_MONTH_PERCENT_CHANGE	number		The percentage change between current price and price from one month ago.
OPEN	number	currency	Indicates the first eligible trade of the day during trading hours. It is set by exchange rules and can be a traded price auction price or an alternative price display.

ID	Type	Supports Conversion	Description
OPEN_INTEREST	number		Open interest is the total number of open or outstanding not closed or delivered options and/or futures contracts that exist on a given day.
PERCENT_CHANGE	number		The percentage change between current price and the previous close price.
PERCENT_VOLUME_THIRTY_DAY_AVERAGE	number		Percentage of day's cumulative volume compared to its thirty day average.
POSTMKT_PRICE	number	currency	Last eligible trade price in post-market session. For US stocks only. Form T and Sold Out of Sequence - Extended Hours are only trades eligible to set the post-market price field.
POSTMKT_VOL	number		Volume executed of last post-market session trade. For US stocks only.
PREMKT_PRICE	number	currency	Last eligible trade price in pre-market session. For US stocks only. Form T and Sold Out of Sequence - Extended Hours are only trades eligible to set the pre-market price field.
PREMKT_VOL	number		Volume executed of last pre-market session trade. For US stocks only.

ID	Type	Supports Conversion	Description
PREVIOUS_CLOSE_PRICE	number		Previous value of the last price on the last day or 2 days prior to a given security traded. This price can be either a traded price or a price a given exchange deems official. This is the most common one for consumers as it can be either adjusted or unadjusted, 1 day or 2 day prior, based on the subscription options. These values can be retrieved directly from given _previousClosePrice* fields
PREVIOUS_S_INTEREST	number		The previous day's short interest value.
PREV_CLOSE	number	currency	Previous value of the last price on the last day a given security traded. This price can be either a traded price or a price a given exchange deems official. The closing price is adjusted for dividends on the ex-date.
PREV_CLOSE_2	number	currency	Closing price from 2 days prior to current trading day T-2.
PREV_CLOSE_DATE	date	timezone	Date in local exchange time of last closing price. Field updates once per day at regional reset.
PREV_CLOSE_TIME	time	timezone	Local Exchange Time in milliseconds of previous

ID	Type	Supports Conversion	Description
			day's or most recent closing price.
PREV_CLOSE_UNADJ	number	currency	Previous value of the last price on the last day a given security traded. This price can be either a traded price or a price a given exchange deems official. The unadjusted closing price is not adjusted for dividends on the ex-date.
PREV_CLOSE_UNADJ_2	number	currency	Unadjusted Closing price from 2 days prior to current trading day T-2.
PREV_SETTLE_DATE	date		Used for Futures: Date of previous settlement.
PRICE_TO_BOOK_RATIO	number		Current price divided by current book value.
PRICE_TO_EARNINGS_RATIO	number		Current price divided by current earnings per share. Calclated as Last / EPS.
PRICE_TO_EARNINGS_RATIO_FISCAL_YEAR_ONE	number		Current price divided by earnings per share of previous fiscal year. Calculated as Last / EPS FY1.
PRICE_VS_VWAP	number		Difference between current price and vwap.
PRICE_VS_VWAP_PERCENTAGE	number		Percentage of current price compared to vwap.
PRODUCT	number		

ID	Type	Supports Conversion	Description
			Indicates product associated with the instrument.
QUARTER_TO_DATE_PERCENT_CHANGE	number		The percentage change between current price and price from the previous quarter.
RATING	string		The Moody's credit rating.
REPORTING_SIDE	string		Indicates which side of the trade (B-Buy or S-Sell) reported the bond sale. Field updates in real time. LSE and TRACE markets only.
RESOLVED_SYMBOL	string		
RESUME	number	currency	First price in session resume price range. Used for pit session from US futures. Pits have been mostly shutdown and limited use.
RESUME_2	number	currency	End price in session resume price range. Used for pit session from US futures. Pits have been mostly shutdown and limited use.
RETURN_12M	number		Twelve-Month total return for mutual funds. Field updates once per day with NAV and distribution update from the Nasdaq Financial Network. Field updates in real time where available else from fund providers (Morningstar, Fundata Canada, FE

ID	Type	Supports Conversion	Description
			Fundinfo, FactSet Pricing Database, FactSet ETF, FactSet Funds) based on the user subscription.
RETURN_1M	number		One month compounded total return for mutual funds. Includes cash and capital distributions. Field updates once per day with NAV and distribution update from the Nasdaq Financial Network. Field updates in real time where available else from fund providers (Morningstar, Fundata Canada, FE Fundinfo, FactSet Pricing Database, FactSet ETF, FactSet Funds) based on the user subscription.
RETURN_3M	number		Three months compounded total return for mutual funds. Field updates once per day with NAV and distribution update from the Nasdaq Financial Network. Field updates in real time where available else from fund providers (Morningstar, Fundata Canada, FE Fundinfo, FactSet Pricing Database, FactSet ETF, FactSet Funds) based on the user subscription.
RETURN_MTD	number		The month-to-date mutual fund return as provided from the real-time server (where available) else from fund providers (Morningstar,

ID	Type	Supports Conversion	Description
			Fundata Canada, FE Fundinfo, FactSet Pricing Database, FactSet ETF, FactSet Funds) based on the user subscription.
RETURN_QTD	number		The quarter-to-date mutual fund return as provided from the real-time server (where available) else from fund providers (Morningstar, Fundata Canada, FE Fundinfo, FactSet Pricing Database, FactSet ETF, FactSet Funds) based on the user subscription.
RETURN_WTD	number		The week-to-date mutual fund return as provided from the real-time server (where available) else from fund providers (Morningstar, Fundata Canada, FE Fundinfo, FactSet Pricing Database, FactSet ETF, FactSet Funds) based on the user subscription.
RETURN_YTD	number		The year-to-date mutual fund return as provided from the real-time server (where available) else from fund providers (Morningstar, Fundata Canada, FE Fundinfo, FactSet Pricing Database, FactSet ETF, FactSet Funds) based on the user subscription.
SECURITY_STATUS	number		Indicates the trading status of a given security. Possible

ID	Type	Supports Conversion	Description
			values are: 0 - Actively trading no restrictions; 2 - Regulatory Halt; 3 - Trade Resumption; 20 - Market Closed; 25 - Inactive Security.
SELLER_DESCRIPTION	string		
SELL_ID	string		ID for broker on sell side of the trade.
SETTLEMENT	number	currency	For Derivatives: The settlement closing price at the end of a trading day. This price includes all trading sessions. The following day's dollar and percent change figures are calculated from the latest settlement price.
SETTLEMENT_DATE	date		Date of the last settlement price.
SETTLEMENT_INDICATOR	number		Indicates whether a future has settled for the day. Values are: 1 - Settled; 0 - No Settlement.
S_INTEREST	number		The quantity of stocks shares investors have sold short, but have yet to cover or close out.
S_INTEREST_CHANGE	number		Change in Short Interest.
THREE_MONTH_PERCENT_CHANGE	number		

ID	Type	Supports Conversion	Description
			The percentage change between current price and price from three months ago.
TICK	string		Indicates the tick direction. The up/down colors indicate whether the last tick is up/ down from the previous tick. Up/Down Arrow - Real-time streaming. D - Delayed streaming. E - End-of-day pricing (from Interactive and Global Prices). F - Prices are from an OFDB file. H - Halted security.
TRADED	number		Indicates if an instrument has traded for the day. 1 indicates the instrument has traded.
TRADED_CONDITION	string		Trade condition of last traded price.
TRADED_DATE	date	timezone	Local date for last trade.
TRADED_PRICE	number	currency	Indicates the last traded price - whether eligible or ineligible - for trades occurring in the main trading session of a given exchange. Determined by trade condition the last traded price may differ from the Last Price field. The last traded price field does not contain the official closing price if disseminated by the exchange. It supports only trades with volume including auctions.

ID	Type	Supports Conversion	Description
TRADED_TIME	time	timezone	Local time for last trade.
TRADED_VOL	number		Last trade volume.
TRADED_YIELD	number		Yield on a bond associated with last traded price.
TRADE_CONDITION	string		Code to indicate conditions information status of a given trade. There can be multiple conditions per trade. Application translates to a alpha code. Trade conditions are based on exchange rules convention sand processing.
TRADE_COUNT	number		Number of trades during the trading day. All trades, eligible or ineligible, are included in the calculation.
TRADE_DATE	date		Date, in local exchange time, of the last price update or eligible activity. Field updates with first eligible update of the day.
TWELVE_MONTH_PERCENT_CHANGE	number		The percentage change between current price and price from twelve months ago.
UNCROSSING_PRICE	number	currency	Auction Price. Known on the LSE as uncrossing price. Mean price at which the bids and asks can be matched.
UNCROSSING_VOL	number		Volume of the auction. Field will display indicative auction

ID	Type	Supports Conversion	Description
			volumes where the exchange disseminates.
UNDERLYING_SECURITY	string		Market Movers: Displays symbol of associated instrument for Market Movers calculations. Futures: Underlying symbol of the exchange contract. Used for underlying symbol in messages for Block Trade Monitor.
VENDOR_SYMBOL	string		Ticker symbol of the instrument.
VWAP	number	currency	Real time Volume-Weighted Average Price VWAP. Calculated as price multiplied by number of shares traded divided by the total shares traded for the day. Displays either exchange provided or FDS-calculated VWAP. Not all trades are included in VWAP definition.
WEEK_TO_DATE_PERCENT_CHANGE	number		The percentage change between current price and ending price of the previous week.
YEAR_TO_DATE_PERCENT_CHANGE	number		The percentage change between current price and ending price of the previous year.
YIELD_TO_MATURITY	number		The total anticipated return on a bond that is held until maturity.

ID	Type	Supports Conversion	Description