IRIS API Documentation

Full description of IRIS Program Interfaces

Version 2.24.0
27.06.2024

Content

1. Introduction

1.1. Scope of document

The document describes application interfaces to obtain information from IRIS IS (hereinafter IRIS API). The description has been prepared on the hypothesis that IRIS API will act as a main source of data for all KASE applications associated with distribution of verified exchange trading information.

1.2. Interaction with IRIS IS

IRIS API module is an export module for IRIS IS. The module outputs data from the IRIS storage. To work with data in the storage, the module uses the internal IRIS CORE software interfaces and the IRIS data bus (see Figure 1, IRIS IS Diagram).

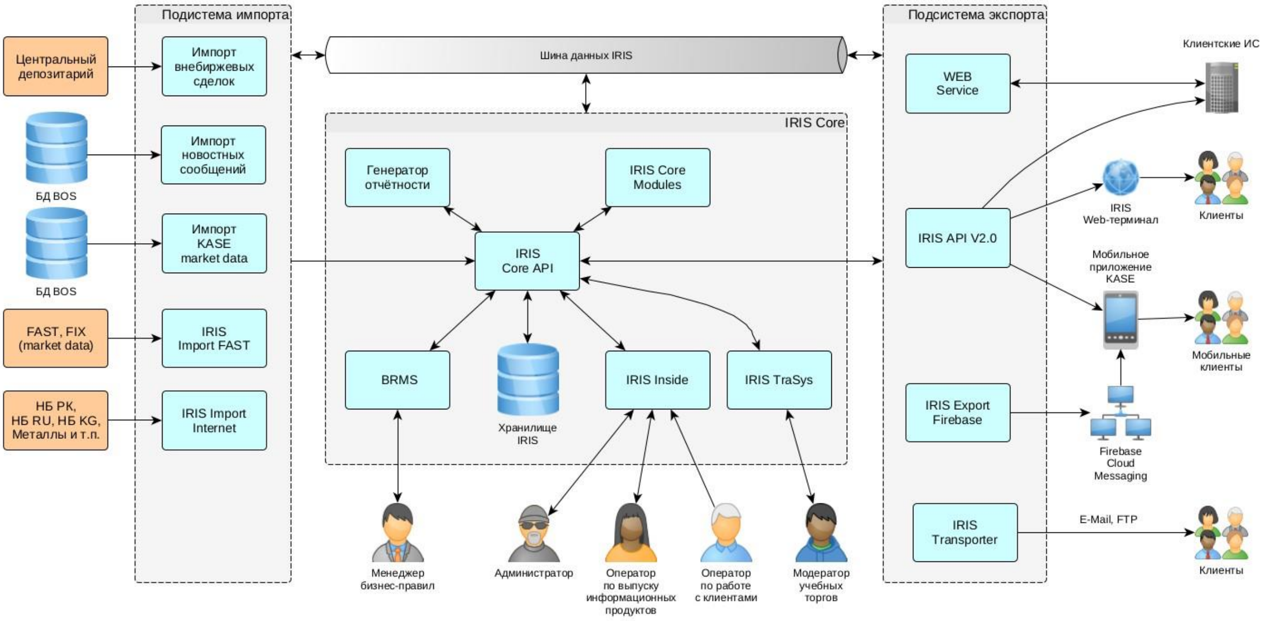


Figure 1. Diagram IRIS IS

When executing requests to IRIS API, user access rights are verified using the IRIS client base. Access rights and limitations are managed in IRIS Inside module.

Currently, IRIS API is used in such new projects as IRIS Web terminal and KASE mobile application.

It is planned that in the future IRIS API module will be expanded to support operation of a client part of the training trading subsystem. Administrative part of the subsystem will be based on developments of the TCS Admin Console project. The corresponding project has been assigned code name KASE TraSys (KASE Training Trading System).

1.3. Description of IRIS API Export Module

IRIS API export module provides user friendly, fully asynchronous, software interfaces that can be accessed from various systems and using a wide range of programming languages (see Figure 2, IRIS API export module diagram).

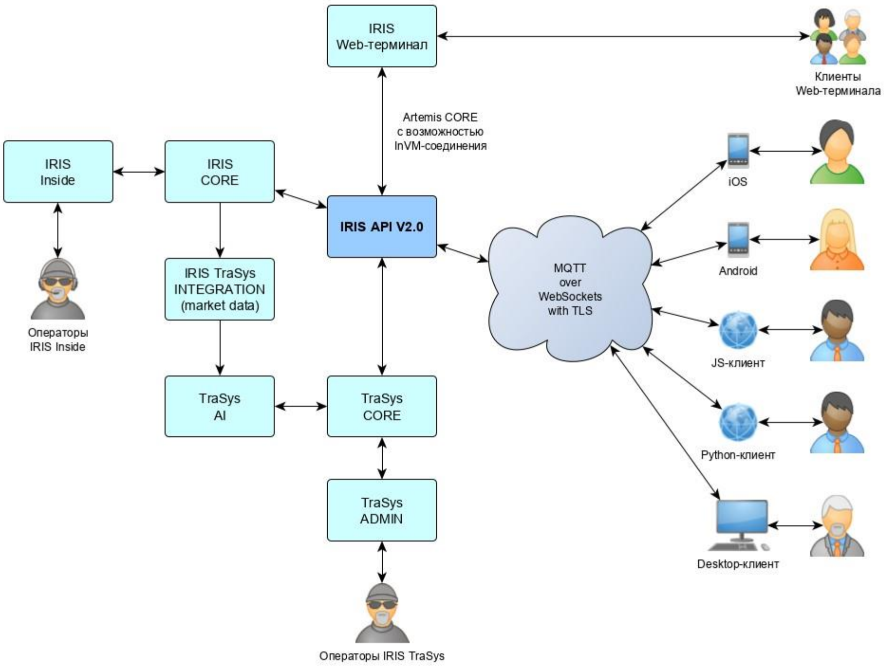


Figure 2. IRIS API export module diagram

IRIS API uses `MQTT`[1][1] protocol as a protocol – a simplified network protocol designed for exchange of messages between devices based on the publisher-subscriber principle. The used MQTT 3.1.1 specification was standardized by the OASIS consortium in 2014.

The protocol focuses on ease of use, low load on communication channels, operation in conditions of constant loss of communication, and easy integration into any system.

The protocol does not impose limitations on format of the transmitted data. Within IRIS API, transmitted messages are serialized and deserialized using the `Protocol Buffers`[2] library, which is also available for a large list of programming languages and is currently a de facto standard in the field of serialization (transfer) of structured data. Protocol Buffers is an efficient binary alternative to the XML text format. Protocol Buffers is simpler, more compact and faster than XML, since binary data are transmitted optimized for a minimum message size. Transmitted IRIS API messages are described below (see Section 5, IRIS API Messages).

In IRIS API, the MQTT protocol is used over the WebSockets protocol using the TLS transport layer protocol to protect data. Using WebSockets simplifies the use of IRIS API from internal networks of organizations with additional protection, since the connection to the IRIS servers is initiated by a regular HTTPS request.

In the current implementation, the TLS connection to IRIS API is terminated on the Nginx web server (see Figure 3, IRIS API Server Infrastructure). The Nginx server uses `Let’s Encrypt`[3] certificates.

Apache ActiveMQ Artemis[4] is used as an MQTT message broker - a universal message broker built into the WildFly application server (version 15 at the time of writing)..

Artemis has an extensible protocol stack. MQTT protocol included in standard Artemis delivery has been improved as part of IRIS API project to provide better integration with the IRIS export subsystem

[1] http://mqtt.org/

[2] https://developers.google.com/protocol-buffers/

[3] https://letsencrypt.org/

[4] http://activemq.apache.org/artemis/

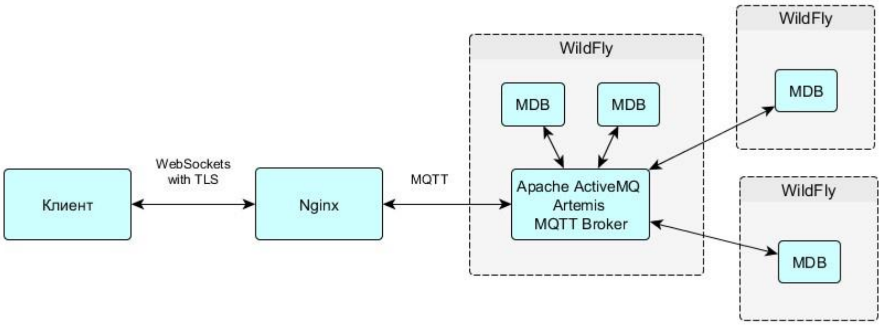


Figure 3. IRIS API server infrastructure

Messages received by broker are sent to IRIS data bus and can be processed asynchronously with the ability to provide load balancing transparent to IRIS API submodules.

1.4. IRIS API as a REST Service

Some of IRIS API methods are provided via REST service. Documentation on REST service methods is available in the `OpenAPI`[5] specification format at https://irisapi.kase.kz/irisopenapi.

The REST-service provides methods in the following sections:

- Market valuation information (MARKET_VALUATION);
- Representative list information (REPRESENTATIVE_LIST);
- Issuer information (ISSUERS);
- User registration (REGISTRATION);
- Project "Treading line" (INFO);

1.4.1. Information about Market Valuation

IRIS API methods for work with market valuation data:

Loading market valuation	The method returns a list of market valuation records in JSON format, limited by the required date. If there is no market valuation for a specified date, the method returns an empty list. If no date is specified, the method will return data about the last valuation made.

1.4.2. Information About Representative List

IRIS API methods to work with data about a representative list of market indices of government securities with a maturity of 31 days or more (KZGB_CPm1m, KZGB_DPm1m, KZGB_Ym1m):

Loading the representative list	The method returns a list of records with data about a representative list in JSON format, limited by the required list modification date. If there is no representative list for a specified date, the method returns an empty list. If no date is specified, the method will return data about the latest version

[5] https://github.com/OAI/OpenAPI-Specification/blob/main/versions/3.0.3.md

	of the representative list.

1.4.3. Information About Issuers

IRIS API methods to work with issuer information:

Loading issuer attributes by code	The method returns issuer attributes by code in JSON format. Format of the returned JSON file is described in the reference tables section (see Section 4.1, "Structure of a JSON file with issuer attributes"). The issuer code (for example, KZTO) and the required reply language are specified as parameters when calling the method.

1.4.4. Registration of Users

IRIS API methods for user registration (for more details, see Section 3.3.1, "User Registration"):

Getting a captcha	Getting a captcha used to check for a robot. Sending registration information.
Registration	The user goes to the registration form and enters their data. The data entered in the form is sent to the server.

Confirmation of registration	Registration confirmation via a link from an e-mail. The user follows the link from the letter. The server finds a previously registered user in the IRIS database, activates them and grants them rights according to the list established for this type of registration.

1.4.5. "Crawline" Project

As part of the "Crawline" project, a number of methods to obtain derivative trading information were added to the REST service:

KASE Index Securities	indkasequotes method. Quotes of securities included in the KASE Index list.
Securities (graph)	secgraph method. Price change graph for an instrument specified in parameters.
Foreign currency quotes	curquotes method. Foreign currencies with prices for the last successful trading session with a sampling depth limit of 10 trading days.
Foreign currencies (graph)	curgraph method. Price change graph for the foreign currency specified in the parameters.

Indicators	indquotes method. The latest indicator values from a fixed list.
Indicators (graph)	indgraph method. Graph of the indicator values specified in the method parameters.
All quotes	allquotes method. The method returns a single list of prices for the last successful trading session on the stock and currency markets, as well as the latest indicator values. In fact, the results of a single request combine the results issued by by the indkasequotes, curquotes and indquotes methods.
KASE Global TOP-20	globaltop20 method. 20 securities among KASE Global shares with the largest count of deals for the current session (company, ticker, price and trend).
Trades schedule	schedule method. The method returns the trades schedule according to concepts described in detail in the relevant section of the documentation (see Section 3.2.1, "Trades Schedule"). This data model appeared after lengthy discussions with the regulations for the mobile application.
Simplified schedule	tradessimpleschedule method. To simplify requests according to the schedule, this method returns only five attributes for each market (start - trades start time, end - trades end time, marketid - market code, market - market name, description - "submarket" description, submarketid - "submarket" code).
Offering announcements	offeringsAdvt method. Announcements of upcoming offerings for the market specified in the parameters. Data source - KASE news messages.
Offering results	offeringsResults method. Results of past offerings for the market specified in the parameters. Data source - KASE news messages.

2. Description of the Development Process

Description of the development process is in the version of the document for internal use. To obtain the document, send a request to v.pyankov@kase.kz.

3. Use of IRIS API

At the application level, using IRIS API consists of performing the following actions:

- connecting to IRIS API server
- subscribing to the required outgoing message queues
- sending messages to the incoming message queues

Each connection to the server is characterized by two main parameters - a user name and a client identifier. Several connections with the same user name can be created at the same time. In this case, each of these connections must have a unique client identifier. An attempt to connect with the same client identifier will cause an error.

When working with IRIS API, three types of queues are used:

- incoming queues used to send request messages

- outgoing queues intended for sending messages to a subscriber with a unique client identifier
- outgoing queues intended for sending broadcast messages received by all subscribers regardless of the client identifier

The approach described above can be illustrated by the following simplified code in Java (Paho library is used):

```
MqttClient client = new MqttClient(url, clientId);
❶client.setCallback(new MqttCallback() { ❷   @Override    public
void messageArrived(String topic, MqttMessage message) {
Reply reply = Reply.parseFrom(message.getPayload());
      ... ❸
   }
});

MqttConnectOptions opts = new MqttConnectOptions();

opts.setUserName(userName);

opts.setPassword(password); client.connect(opts);

❹client.subscribe(inputTopic); ❺

Request reqv = Request.newBuilder()
                     .set(...) ❻
                     .build();; MqttTopic outputTopic =

client.getTopic(output); outputTopic.publish(new

MqttMessage(reqv.toByteArray())); ❼
```

❶ A client is created with a unique client ID.
❷ An incoming message handler is registered for the client.
❸ The message arrives as a byte array and is deserialized using the Protocol Buffers library.
❹ A connection to the server is created. Parameters are specified for each connection. At a minimum, this is the username and password. Additionally, parameters related to, for example, the TLS protocol can be specified.
❺ Subscription to the required queue. After subscribtion, messages will come in the previously created handler. The queue can be determined by the "topic" parameter.
❻ Message to be sent is created using the Protocol Buffers library.
❼ Message is serialized using the Protocol Buffers library into a byte array and published to the required queue.

It must be noted that value of the primary key in IRIS cannot be less than one. It follows that if the value "0" was received in fields with the **int32** or int64 **type**, then it must be perceived as NULL. This is consistent with the inability to pass NULL values in **int32** or **int64** fields in Protocol Buffers.

3.1. Open Information

IRIS API section for work with general information, access to which does not require authorization. To work with the section, you must connect to the server using the fixed user name iris@open.kase.kz and the password free.

When working with IRIS API in the open information section, two message queues are used:

- jms/topic/open/iris/Info - a queue to send request messages (see Section 5.1.27, "OpenInfoApiRequest")

- jms/topic/open/iris/Info/client - a queue to receive messages by subscribers with a unique client identifier (see Section 5.1.26, "OpenInfoApiReply").

The following types of requests are available:

- user registration request (see Section 5.23.6, "UserRegRequest")
- user password reset request (see Section 5.23.3, "ResetPwdRequest")

3.2. General information

When working with IRIS API in terms of receiving general information, two message queues are used:

- jms/queue/iris/Main - a queue for sending request messages (see Section 5.1.21, "InfoApiRequest")
- jms/topic/iris/Main/client - a queue for receiving messages by subscribers with a unique client identifier (see Section 5.1.20, "InfoApiReply").

The following types of requests are currently supported:

- request for information about the current state of the lists of instruments and indicators recommended for display to the end user (see Section 5.3.14, "MainListsRequest")
- request for a list of active polls (see Section 5.3.22, "PoolsListRequest")
- request for questions from a selected poll (see Section 5.3.20, “PoolQuestionsRequest”)
- rrequest to send answers to a survey (see Section 5.3.17, “PoolAnswersRequest”)
- request to change a user’s password (see Section 5.3.12, “ChangePwdRequest”)
- request for reference data about MOEX modes (see Section 5.3.4, “BoardsRequest”)
- request for information about an IRIS user (see Section 5.3.35, “UserInfoRequest”)
- message to be sent by the user to provide feedback (see Section 5.3.32, “UserFeedbackRequest”)
- request for information about a trades schedule on the KASE exchange (see Section 5.3.28, “TradesScheduleRequest”)
- request for information about a status of trades on the KASE exchange (see Section 5.3.30, “TradesStatusRequest”)
- request to receive a list of brokers (see Section 5.3.10, “BrokersListRequest”)
- request to be sent by a user to a broker (see Section 5.3.7, “BrokerMessageRequest”)

3.2.1. Trades Regulation

3.2.1.1. Basic Notions

Terms used in the description of the data model used to transmit information about trades schedule:

Market	Trading of a group of exchange instruments organized in a certain way. For example, the stock market, the bond market, the currency market, etc. Usually, in applications or on the website, trading information is grouped by markets. For example, in the KASE mobile application, each market has its own tab.
Part of the market	A group of instruments related to one market, but having different trading times or regulations.
Trading time	A period of time during which trading takes place on a certain part of the market during a trading day. Determined by the start time and end time.
Session	A period of time during a trading day with a start time, end time, and a free text description. Session can relate to both the market as a whole and to a separate part of the market. Start time and end time of the session cannot go beyond the trading time on the market (or on a part of the market) to which this session relates.
Trades schedule	A list of all sessions during a trading day on ta market or on a separate part of the market. In other words, the trades schedule relate to a market or a part of the market. We say: "Stock Market Trades Schedule", "KASE Global Stock Market Trades Schedule" or "CNY/KZT Trades Schedule with T+0 Settlement".
Trades Status	Current trades status for the selected instrument.

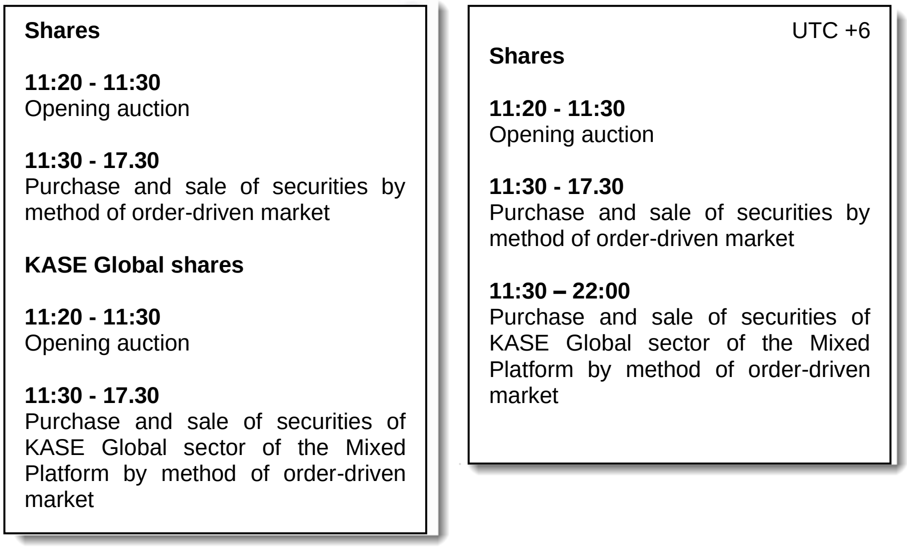
3.2.1.2. Use Options

Let's study following options for using the information about trades schedule:

- displaying the trades schedule for the market;
- displaying the trades status for the selected instrument;

To obtain information about trades schedule, you must execute a TradesScheduleRequest request and receive a TradesScheduleReply message with a list of markets in the `markets` field. The required market is found in the `market` field in MarketRecord. If information is required for only one specific market, it must be specified in the market field of the TradesScheduleRequest request.

Examples of displaying the trades schedule can be seen below (see Figure 4, Examples of displaying the trades schedule):



Shares

11:20 - 11:30
Opening auction

11:30 - 17.30
Purchase and sale of securities by method of order-driven market

KASE Global shares

11:20 - 11:30
Opening auction

11:30 - 17.30
Purchase and sale of securities of KASE Global sector of the Mixed Platform by method of order-driven market

UTC +6
Shares

11:20 - 11:30
Opening auction

11:30 - 17.30
Purchase and sale of securities by method of order-driven market

11:30 – 22:00
Purchase and sale of securities of KASE Global sector of the Mixed Platform by method of order-driven market

Figure 4. Examples of displaying the trades schedule

If the market is not divided into separate parts, the submarkets list from MarketRecord will contain only one record. In a more complex version, the trades schedule can be formed from separate tables for each part of the market. The name field from SubmarketRecord is used as headers. The start and end times of the session are in the `session_start` and `session_end` fields, and description of the session is in the `description` field from SessionRecord.

To obtain information about trades status, it is necessary to execute the TradesStatusRequest request and receive the TradesStatusReply message in reply. In the received reply, is necessary to check the value of the `trading_day` field. If the current day is not a trading day, then the `next_trading_day` field is used to display the next trading day. Otherwise, the status of trades within the current trading day is determined by the `trades_start` and `trades_end` fields and the current time of day.

3.3. Users

The following procedures are supported in IRIS API when working with users:

- initial registration of a new user;
- obtaining information about the user status;
- changing the password by an existing user;
- restoring the password by an existing user.

3.3.1. User Registration

Initial registration of a new user occurs using a form located on one of the pages of the web terminal or on a specially designed landing page. The registration process consists of several steps, shown in the diagram (see Figure 5, User Registration).

Registration steps:

1. The user goes to the registration form and enters its data.

2. Data entered in the form is sent to the server.

3. Data is saved in the IRIS DB. At this point, a client is created in IRIS as an individual with the status of "Potential Client" in a blocked state.

4. A registration letter with a specially designed link is sent to the user's e-mail.

5. The user follows the link in the letter.

6. The server finds a previously registered user in the IRIS database, activates him and grants him rights according to the list established for this type of registration.

7. The user is redirected to the registration confirmation page or to the page with the error description.

8. The administrator sees the registered user in IRIS Inside and takes elevant actions.

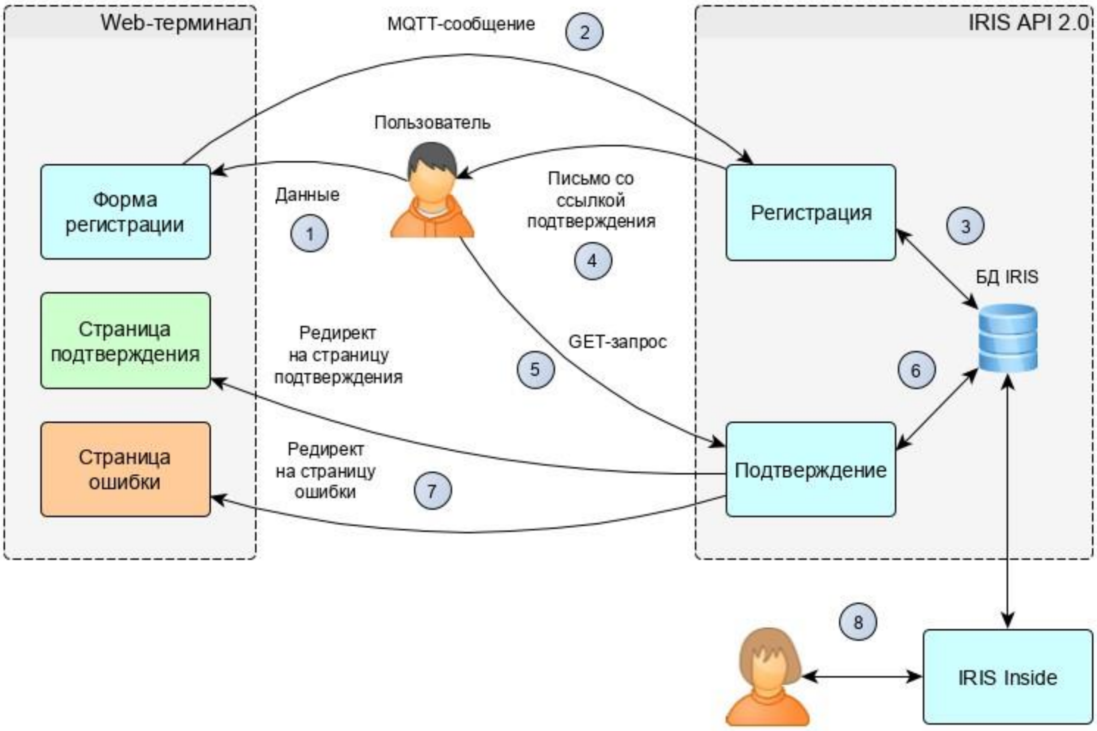


Figure 5. User registration

Using the registration form, the user enters the following data:

- username;
- password;
- full name;
- email address;
- phone number (optional);

- address (optional);
- comments (optional);
- verification code (captcha).

At this stage, IRIS API does not provide support for the payment process. Payment support can be added to IRIS API in the future as a separate process.

Also, IRIS API does not allow linking a user to a legal entity during the registration. Such linking can be done by the operator in the internal administrative module "IRIS Inside" after the user is checked.

3.3.2. User Status

Obtaining information about the user status is performed using a request made under the user name and password specified during the registration. As a reply to the request, all the information entered during the registration is provided, as well as the current user status, for example, "E-mail confirmation failed", "User blocked" or "Password change required".

Status information can be used by the application to inform the user, for example, about the need to change the password.

3.3.3. Password Change

The password is changed in the user profile (see Figure 6, Changing the password) after logging into the application.

Steps to change the password:

1. The user enters a new password.
2. The entered password is sent to the server (see Section 3.2, “General Information” and Section 5.3.12, “ChangePwdRequest”).
3. The new password is saved in the IRIS DB.

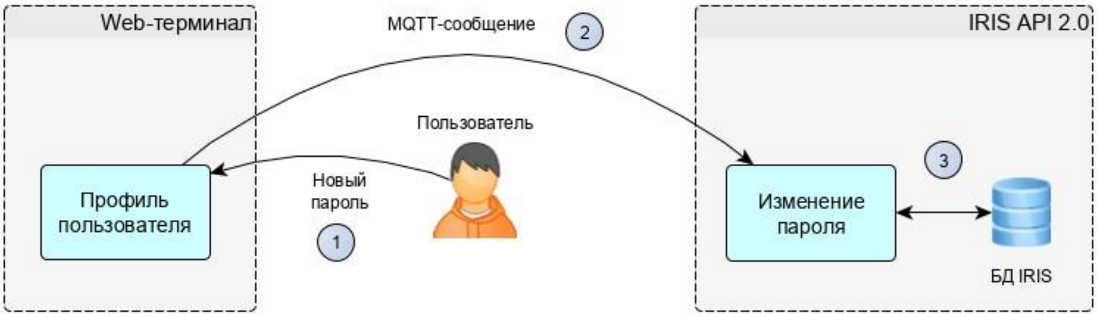


Figure 6. Password change

3.3.4. Password Recovery

The password is recovered by the existing user on the application login form (see Figure 7, Password recovery).

Steps to recover the password:

1. The user enters the login and makes a request to recover the password. For example, by clicking the button or link in the login form.

2. The request is sent to the server. In case of an error during the recovery process, a corresponding message is sent in reply to the request, which is displayed in the login form.

3. The server checks the IRIS database for a user with the specified name. If the user exists, its password is changed to a new, randomly generated one.

4. A new password is sent to the user's e-mail.

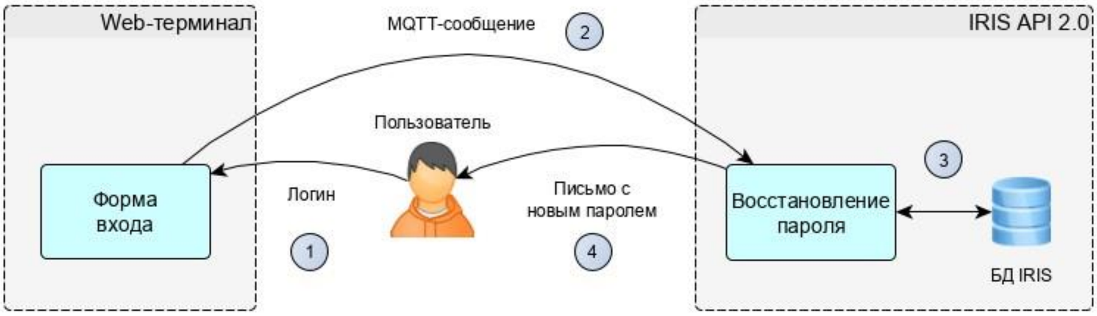


Figure 7. Password recovery

After password recovery, the user can log in to its profile and change the password sent to them at their discretion (see Section 3.3.3, “Changing Password”).

3.4. Stock Market Deals

When working with IRIS API in the context of securities deals, three message queues are used:

- jms/queue/iris/Deals - a queue for sending request messages (see Section 5.1.15, “DealsApiRequest”)

- jms/topic/iris/Deals/client - a queue for receiving messages by subscribers with a unique client identifier (see Section 5.1.14, “DealsApiReply”).

- jms/topic/iris/Deals/broadcast - a queue for receiving broadcast messages (see Section 5.4.11, "DealsUpdate")

There are two main types of request:

- request for a count of deals (see Section 5.4.4, "DealsCountRequest")

- request for a list of deals (see Section 5.4.9, "DealsRequest").

- request for information about deals as a file in a specified format (see Section 5.4.6, "DealsExportRequest").

When executing a request, filtering conditions (see Section 5.4.7, "DealsFilter") and sorting conditions (see Section 5.4.14, "DealsSortField") can be specified.

Filtering can be performed given versioning of records in the IRIS storage. This makes possible to organize a mode of operation with IRIS API in which it is possible to download only new or changed deal records.

The request can contain an instruction to limit the returned result to the required range (see Section 5.2.17, "Range") or the size of the received record package.

DealsUpdate broadcast messages by definition do not contain the instrument code, since it is anonymized information. To receive full information about deals in reply to the DealsUpdate message, it is necessary to send a request on your own behalf for a list of deals with a filter of deal codes and receive full information in reply to the "jms.topic.iris.Deals.client" queue. Full information will be received only if it is available to the user.

Reference tables with codes used when transmitting deal data are located in a separate section of this document (see Section 4, "Reference Tables").

3.5. Stock Market Orders

When working with IRIS API in terms of stock market orders, three message queues are used:

- jms/queue/iris/Orders - a queue for sending request messages (see Section 5.1.31, "OrdersApiRequest")
- jms/topic/iris/Orders/client - a queue for receiving messages by subscribers with a unique client identifier (see Section 5.1.30, "OrdersApiReply").
- jms/topic/iris/Orders/broadcast - a queue for receiving broadcast messages (see Section 5.5.10, "OrdersUpdate")

There are two main types of request:

- request for a count of orders (see Section 5.5.3, "OrdersCountRequest")
- request to receive a list of orders (see Section 5.5.8, "OrdersRequest").

When executing a request, filtering conditions (see Section 5.5.6, "OrdersFilter") and sorting conditions (see Section 5.5.13, "OrdersSortField") can be specified.

Filtering can be performed given versioning of records in the IRIS repository. This makes it possible to organize a mode of working with IRIS API in which it is possible to download only new or changed order records.

The request can contain an instruction to limit the returned result to the required range (see Section 5.2.17, "Range") or the size of the received record package.

OrdersUpdate broadcast messages by definition do not contain the tool code, since it is anonymized information. To receive full information about orders in reply to the OrdersUpdate message, it is necessary send a request on your own behalf for a list of orders with a filter from the order codes and receive full information in reply to the "jms.topic.iris.Orders.client" queue. Full information will be received only if it is available to the user.

3.6. Stock Market Trading Results

When working with IRIS API in terms of transmitting information about trades, grouped by period, the following message queues are used:

- jms/queue/iris/Totals - a queue for sending request messages (see Section 5.1.41, "TotalsApiRequest")
- jms/topic/iris/Totals/client - a queue for receiving messages by subscribers with a unique client identifier (see Section 5.1.40, "TotalsApiReply").
- jms/topic/iris/Totals/broadcast - a queue for receiving broadcast messages in terms of stock market trading results (see Section 5.8.13, "TotalsUpdate").

There are three main types of request:

- request for a count of records with grouped information (see Section 5.8.7, "TotalsCountRequest");

- request to receive a list of records with grouped information (see Section 5.8.12, “TotalsRequest”);
- request to receive quotes (see Section 5.8.2, “QuotationsRequest”);
- request to receive information about the leaders of rise and fall (see Section 5.8.4, “RisersFallersRequest”);
- request to receive information about trading results as a file in the specified format (see Section 5.8.9, “TotalsExportRequest”).

When executing a request, filtering conditions for TotalsRequest, QuotationsRequest and TotalsExportRequest must be specified (see Section 5.8.10, “TotalsFilter” and Section 5.4.2, “DealsBaseFilter”). If the conditions are not specified, an error message will be received in reply to the request.

Trading results do not include data about over-the-counter deals. As a result, including the OTC deal type in the filtering conditions (see Section 5.4.13, “DealType”) does not make sense with respect to the trading results on the stock market.

TotalsRequest request can contain an instruction to limit the returned result to the required range (see Section 5.2.17, “Range”) or to the size of the received record package.

TotalsUpdate message arriving in the broadcast queue must be used to update trading results, quotes, and information about rise and fall leaders.

3.7. Order Book on the Stock Market

The order book is a table of orders to buy and sell securities on the stock market. The order book displays the total count of pending orders to buy and sell securities at each price above and below the market price.

When working with IRIS API in terms of displaying the order book on the securities market, the following message queues are used:

- jms/queue/iris/Orderbook - a queue for sending request messages (see Section 5.1.29, "OrderbookApiRequest")
- jms/topic/iris/Orderbook/client - a queue for receiving messages by subscribers with a unique client identifier (see Section 5.1.28, "OrderbookApiReply").
- jms/topic/iris/Orderbook/broadcast - a queue for receiving broadcast messages (see Section 5.14.3, "OrderbookUpdate")

There is one type of request for an order book:

- request to receive the order book (see Section 5.1.29, "OrderbookApiRequest").

OrderbookUpdate broadcast messages contain only the instrument code, since it is anonymized information. To receive full information about the order book, in reply to the OrderbookUpdate message, it is necessary to send an OrderbookApiRequest request on your behalf indicating the instrument code. Full information will be received only if it is available to the user.

In the current version of IRIS API, the same limitations apply to work with the order book as when working with aggregated data about the stock and currency markets (see Section 3.6, "Stock Market Trading Results" and Section 3.11, "Currency Market Trading Results").

One message to the queue "jms/topic/iris/Orderbook/broadcast" can arrive in reply to the processing of several orders at once (depending on the timeout settings on the server).

3.8. Stock Market Graphs

When working with IRIS API in terms of receiving data for plotting instrument price graphs and trading volumes, the following message queues are used:

- jms/queue/iris/Graph - a queue for sending request messages (see Section 5.1.17, "GraphApiRequest")
- jms/topic/iris/Graph/client - a queue for receiving messages by subscribers with a unique client identifier (see Section 5.1.16, "GraphApiReply").

Streaming data updates (streaming data updates) is not supported in the current version of IRIS API.

When working with data for plotting graphs, there are two types of requests:

- request to receive data to plot graphs (see Section 5.16.2, "GraphRequest").
- request to receive data to plot a heat map (see Section 5.16.5, “HeatmapRequest”).

When performing a request to receive data to plot graphs, the instrument code must be specified (see Section 5.16.2, “GraphRequest”). If the period is not specified in the request (see the period field in GraphRequest), the reply will be a graph for the entire period of trading on the stock market (starting from September 1997).

Interval between readings is selected automatically, depending on the period specified in the request. The selected interval is returned in the interval field of the GraphReply message.

Heat map (see Section 5.16.5, “HeatmapRequest”) is a graphical representation of data, where individual values in a table are displayed using a color.

In our case, we use three values: security code, issuer capitalization, trend:

1. Security code is published in each cell of the map;
2. Issuer capitalization affects the size of each cell - the higher the capitalization, the proportionally larger the cell;
3. Trend, in % - it is a ratio of the last price (or closing price at the time of closing of the trading session) of the current/last successful session to the last price/closing price of the previous successful session;
4. Cell color depends on the trend - if the trend is positive, the cell must be green, if negative, the cell turns red, if the trend is zero, the cell must turn gray;

Two heat map options are supported (see Section 5.16.7, "HeatmapType"):

1. KASE Index shares;
2. Foreign shares of the KASE Global platform;

3.9. Currency Market Deals

When working with IRIS API in terms of currency market deals, three message queues are used:

- jms/queue/iris/CurDeals - a queue for sending request messages (see Section 5.1.5, "CurDealsApiRequest")
- jms/topic/iris/CurDeals/client - a queue for receiving messages by subscribers with a unique client identifier (see Section 5.1.4, "CurDealsApiReply").
- jms/topic/iris/CurDeals/broadcast - a queue for receiving broadcast messages (see Section 5.11.10, "CurDealsUpdate")

There are two main types of request:

- request for a count of deals (see Section 5.11.3, "CurDealsCountRequest")
- request to receive a list of deals (see Section 5.11.8, "CurDealsRequest").

When executing a request, filtering conditions (see Section 5.11.6, "CurDealsFilter") and sorting conditions (see Section 5.11.12, "CurDealsSortField") can be specified.

Filtering can be performed given versioning of records in the IRIS storage. This allows makes it possible to organize a mode of working with IRIS API in which it is possible to download only new or changed deal records.

The request can contain an instruction to limit the returned result to the required range (see Section 5.2.17, "Range") or the size of the received record package.

CurDealsUpdate broadcast messages by definition do not contain the instrument code, since it is anonymized information. To receive full information about deals in reply to the CurDealsUpdate message, it is necessary to send a request on your behalf for a list of deals with a filter from deal codes and receive full information in reply to the "jms.topic.iris.CurDeals.client" queue. Full information will be received only if it is available to the user.

3.10. Currency Market Orders

When working with IRIS API in terms of currency market orders, three message queues are used:

- jms/queue/iris/CurOrders - a queue for sending request messages (see Section 5.1.11, "CurOrdersApiRequest")
- jms/topic/iris/CurOrders/client - a queue for receiving messages by subscribers with a unique client identifier (see Section 5.1.10, "CurOrdersApiReply").
- jms/topic/iris/CurOrders/broadcast - a queue for receiving broadcast messages (see Section 5.13.10, "CurOrdersUpdate")

There are two main types of requests:

- request for a count of orders on the currency market (see Section 5.13.3, "CurOrdersCountRequest")
- request to receive a list of orders on the currency market (see Section 5.13.8, "CurOrdersRequest").

When executing a request, filtering conditions (see Section 5.13.6, "CurOrdersFilter") and sorting (see Section 5.13.12, "CurOrdersSortField") can be specified.

Filtering can be performed given versioning of records in the IRIS storage. This makes it possible to organize a mode of work with IRIS API in which it is possible to download only new or changed order records.

The request can contain an instruction to limit the returned result to the required range (see Section 5.2.17, "Range") or to the size of the received record package.

CurOrdersUpdate broadcast messages by definition do not contain the instrument code, since it is anonymized information. To obtain full information about requests in reply to the CurOrdersUpdate message, it is necessary to send a request on your behalf for the request list with a filter from request codes and receive full information in reply to the "jms.topic.iris.CurOrders.client" queue. Full information will be received only if it is available to the user.

3.11. Currency Market Trading Results

When working with IRIS API in terms of transmitting information about currency market trades, grouped by period, the following message queues are used:

- jms/queue/iris/CurTotals - a queue for sending request messages (see Section 5.1.13, "CurTotalsApiRequest")
- jms/topic/iris/CurTotals/client - a queue for receiving messages by subscribers with a unique client identifier (see Section 5.1.12, "CurTotalsApiReply").
- jms/topic/iris/CurTotals/broadcast - a queue for receiving broadcast messages in terms of deals with securities (see Section 5.12.11, "CurTotalsUpdate")

There are three main types of request:

- request for a count of records with grouped information (see Section 5.12.5, "CurTotalsCountRequest")
- request for a list of records with grouped information (see Section 5.12.10, "CurTotalsRequest").
- request for information about quotes on the currency market (see Section 5.12.2, "CurQuotationsRequest").
- request for information about trading results as a file in a specified format (see Section 5.12.7, "CurTotalsExportRequest").

When executing a request, filtering conditions must be specified (see Section 5.12.8, "CurTotalsFilter"). If the conditions are not specified, an error message will be received in reply to the request.

The request can contain an instruction to limit the returned result to the required range (see Section 5.2.17, "Range") or the size of the received record package.

3.12. Order Book on the Currency market

Order book is a table of buy and sell orders on the currency market. Order book displays the total count of pending buy and sell orders at each price above and below the market price.

When working with IRIS API in terms of displaying the order book on the currency market, the following message queues are used:

- jms/queue/iris/CurOrderbook - a queue for sending request messages (see Section 5.1.9, "CurOrderbookApiRequest")
- jms/topic/iris/CurOrderbook/client - a queue for receiving messages by subscribers with a unique client identifier (see Section 5.1.8, "CurOrderbookApiReply").
- jms/topic/iris/CurOrderbook/broadcast - a queue for receiving broadcast messages (see Section 5.15.3, "CurOrderbookUpdate")

There is one type of request for an order book on the currency market:

- request to receive the order book (see Section 5.1.9, "CurOrderbookApiRequest").

Broadcast CurOrderbookUpdate messages contain only the instrument code, since it is anonymized information. To receive full information about the order book, in reply to the CurOrderbookUpdate message, it is necessary to send a CurOrderbookApiRequest request on your behalf indicating the instrument code. Full information will be received only if it is available to the user.

In the current version of IRIS API, the same limitations apply to working with the order book as when working with aggregated data about the stock and currency markets (see Section 3.6, "Stock Market Trading Results" and Section 3.11, "Currency Market Trading Results").

One message to the queue "jms/topic/iris/CurOrderbook/broadcast" can arrive in reply to the processing of several orders at once (depending on the timeout settings on the server).

3.13. Graphs on the Currency Market

When working with IRIS API in terms of receiving data for constructing price graphs and trading volumes on the currency market, the following message queues are used:

- jms/queue/iris/CurGraph - a queue for sending request messages (see Section 5.1.7, "CurGraphApiRequest")
- jms/topic/iris/CurGraph/client - a queue for receiving messages by subscribers with a unique client identifier (see Section 5.1.6, "CurGraphApiReply").

Streaming data updates (streaming data updates) is not supported in the current version of IRIS API.

When working with data for plotting the graphs on the currency market, there is only one type of request:

- request to receive data to plot graphs (see Section 5.1.7, "CurGraphApiRequest").

When executing a request, the instrument code must be specified (see Section 5.17.2, "CurGraphRequest"). If the period is not specified in the request (see the `period` field in CurGraphApiRequest), the reply will be a graph for the entire period of trading on the currency market (starting from December 1998).

The interval between readings is selected automatically, depending on the period specified in the request. The selected interval is returned in the interval field of the GraphReply message.

3.14. Repo Market Deals

When working with IRIS API in terms of repo market deals, three message queues are used:

- jms/queue/iris/Repo - a queue for sending request messages (see Section 5.1.33, "RepoApiRequest")
- jms/topic/iris/Repo/client - a queue for receiving messages by subscribers with a unique client identifier (see Section 5.1.32, "RepoApiReply").
- jms/topic/iris/Repo/broadcast - a queue for receiving broadcast messages (see Section 5.19.11, "RepoUpdate")

There are two main types of request:

- request for a count of repo market deals (see Section 5.19.3, "RepoCountRequest")
- request to receive a list of repo market deals (see Section 5.19.9, "RepoRequest").

When executing a request, filtering conditions (see Section 5.19.6, “RepoFilter”) and sorting conditions (see Section 5.19.13, “RepoSortField”) can be specified.

Filtering can be performed given versioning of records in the IRIS storage. This makes it possible to organize a mode of operation with IRIS API in which it is possible to download only new or changed deal records.

The request can contain an instruction to limit the returned result to the required range (see Section 5.2.17, "Range") or the size of the received record package.

RepoUpdate broadcast messages by definition do not contain the instrument code, since it is anonymized information. To receive full information about deals in reply to a RepoUpdate message, it is necessary to send a request on your behalf for a list of deals with a filter of deal codes and receive full information in reply to the "jms.topic.iris.Repo.client" queue. Full information will be received only if it is available to the user.

Reference tables with codes used when transmitting data about repo market deals are located in a separate section of this document (see Section 4, "Reference Tables").

3.15. Orders in the repo operations market

When working with IRIS API in terms of repo orders, three message queues are used:

- jms/queue/iris/RepoOrders - a queue for sending request messages (see Section 5.1.37, "RepoOrdersApiRequest")
- jms/topic/iris/RepoOrders/client - a queue for receiving messages by subscribers with a unique client identifier (see Section 5.1.36, "RepoOrdersApiReply").
- jms/topic/iris/RepoOrders/broadcast - a queue for receiving broadcast messages (see Section 5.20.10, "RepoOrdersUpdate")

There are two main types of request:

- request for a count of orders (see Section 5.20.3, "RepoOrdersCountRequest")
- request to receive a list of orders (see Section 5.20.8, "RepoOrdersRequest").

When executing a request, filtering conditions (see Section 5.20.6, "RepoOrdersFilter") and sorting (see Section 5.20.12, "RepoOrdersSortField") can be specified.

Filtering can be performed given versioning of records in the IRIS repository. This makes it possible to organize a mode of working with IRIS API in which it is possible to download only new or changed order records.

The request can contain an instruction to limit the returned result to the required range (see Section 5.2.17, "Range") or the size of the received record package.

RepoOrdersUpdate broadcast messages do not contain the tool code by definition, since this is impersonal information. To receive full information about orders in reply to the RepoOrdersUpdate message, it is necessary to send a request on your behalf for a list of orders with a filter from the order codes and receive full information in reply to the "jms.topic.iris.RepoOrders.client" queue. Full information will be received only if it is available to the user.

3.16. Order Book on the Repo Market

When working with IRIS API in terms of displaying the order book on the repo market, the following message queues are used:

- jms/queue/iris/RepoOrderbook - a queue for sending request messages (see Section 5.1.35, "RepoOrderbookApiRequest")
- jms/topic/iris/RepoOrderbook/client - a queue for receiving messages by subscribers with a unique client identifier (see Section 5.1.34, "RepoOrderbookApiReply").
- jms/topic/iris/RepoOrderbook/broadcast - a queue for receiving broadcast messages (see Section 5.21.3, "RepoOrderbookUpdate")

There is one type of request for a book:

- request to receive the order book (see Section 5.1.35, "RepoOrderbookApiRequest").

RepoOrderbookUpdate broadcast messages contain only the instrument code and the trading mode, since it is anonymized information. To receive full information about the order book in reply to the

RepoOrderbookUpdate message, it is necessary to send a RepoOrderbookApiRequest request on your behalf indicating the instrument code and the trading mode. Full information will be received only if it is available to the user.

In the current version of IRIS API, the same limitations apply to working with the order book as when working with deals and orders on the repo market.

One message to the queue "jms/topic/iris/RepoOrderbook/broadcast" can arrive in reply to processing of several requests at once (depending on the throttling settings on the server).

3.17. User Watchlist

When working with a user watchlist, IRIS API uses three message queues:

- jms/queue/iris/Watchlist - a queue for sending request messages (see Section 5.1.43, "WatchlistApiRequest")
- jms/topic/iris/Watchlist/client - a queue for receiving messages by subscribers with a unique client identifier (see Section 5.1.42, "WatchlistApiReply").
- jms/topic/iris/Watchlist/broadcast - a queue for receiving broadcast messages (see Section 5.22.10, “WatchlistUpdate”)

The following requests are available for working with the watchlist:

- request to search for instruments when adding to the list (see Section 5.22.9, “WatchlistSearchRequest”)
- request to add instruments to the list (see Section 5.22.2, “WatchlistAddRequest”).
- request to remove instruments from the list (see Section 5.22.4, “WatchlistDelRequest”).
- request to receive quotes for instruments included in the user’s watchlist (see Section 5.22.7, “WatchlistRequest”).

WatchlistUpdate broadcast message contains a list of user names whose watchlists are subject to updating. If a user name is found in this list, a WatchlistRequest request must be sent to the "jms.topic.iris.Watchlist" queue and a reply must be received in "jms.topic.iris.Watchlist.client".

3.18. News Messages

When working with IRIS API in terms of loading news messages, three queues are used:

- jms/queue/iris/News - a queue for sending request messages (see Section 5.1.25, "NewsApiRequest")
- jms/topic/iris/News/client - a queue for receiving messages by subscribers with a unique client identifier (see Section 5.1.24, "NewsApiReply").
- jms/topic/iris/News/broadcast - a queue for receiving broadcast messages (see Section 5.9.23, "NewsUpdate")

There are nine basic types of requests:

- request for a count of news messages that match the filtering conditions (see Section 5.9.15, "NewsCountRequest")
- request to receive a list of news message titles (see Section 5.9.21, "NewsRequest").
- request to receive news message bodies by their primary keys (see Section 5.9.13, "NewsBodyRequest").

- request to receive news message links by their primary keys (news can be linked to issuers, securities, and a category index, and also have attached files, see Section 5.9.19, "NewsRefsRequest").
- request to receive a list of files attached to a news message by its primary key (Section 5.9.6, “FilesRequest”).
- request to receive content of a file by its primary key (Section 5.9.4, “FileBodyRequest”).
- request to receive full information about news messages. Used to reduce the count of requests to the server (Section 5.9.9, “FullNewsInfoRequest”).
- request to add news messages to the favorites list (Section 5.9.25, “NewsUsersAddRequest”).
- request to delete news messages from the favorites list (Section 5.9.27, “NewsUsersDeleteRequest”).

When executing a request, filtering conditions (see Section 5.9.16, “NewsFilter”) and sorting conditions (see Section 5.9.22, “NewsSortRecord”) can be specified. Messages can be filtered by time (see Section 5.2.3, "DatePeriod"), language, by belonging to specified categories, in relation to securities and issuers. A full-text search condition can be specified for filtering the news. A list of news message codes for which information is required can also be directly specified in the filter.

Filtering can be performed given versioning of records in the IRIS storage. This allows organizing a mode of working with IRIS API in which it is possible to download only new or changed records of news messages.

The request can contain an instruction to limit the returned result to the required range (see Section 5.2.17, "Range") or the size of the received record package.

NewsUpdate broadcast messages contain only the primary key of the record, the language and the time of publication of the news message. To receive full information about the news in reply to the NewsUpdate message, it is necessary to send on your behalf and receive full information in the "jms.topic.iris.News.client" queue. Full information will be received only if it is available to the user.

In this version of IRIS API, messages for working with files are located directly in this section (see Section 5.9.6, "FilesRequest", Section 5.9.4, "FileBodyRequest", Section 5.9.5, "FilesReply" and Section 5.9.3, "FileBodyReply").

3.19. Issuer Attributes

When working with IRIS API in terms of issuer information, three message queues are used:

- jms/queue/iris/Issuers - a queue for sending request messages (see Section 5.1.23, "IssuersApiRequest")
- jms/topic/iris/Issuers/client - a queue for receiving messages by subscribers with a unique client identifier (see Section 5.1.22, "IssuersApiReply").
- jms/topic/iris/Issuers/broadcast - a queue for receiving broadcast messages

Currently, it is possible to execute the following queries:

- request to receive a list of issuers (see Section 5.6.6, "IssuersRequest")
- request to receive issuer attributes (see Section 5.6.3, "IssuerAttrsRequest")

When executing the request to receive a list of issuers, a filtering condition can be specified (see Section 5.6.4, "IssuersFilter").

Filtering can be performed by issuer codes and names. In this case, both full codes and names and their templates can be transmitted (see Section 3.22, "Filtering Templates ").

It is possible to search for an issuer by ISIN of one of its securities. In this case, for older securities, NIN or another possible identifier can be used instead of ISIN.

Filter can explicitly specify a list of issuers' primary keys required for loading.

It is possible to filter by the required record statuses (see Section 5.6.7, "IssuerRecordStatus").

Filtering can also be performed given versioning of records in the IRIS storage. This allows organizing a mode of work with IRIS API in which it is possible to download only new or changed records of issuers.

Request can contain an instruction to limit the returned result to the required range (see Section 5.2.17, "Range") or the size of the received record package.

3.20. Attributes of securities and instruments

When working IRIS API in the context of information about securities and instruments, three message queues are used:

- jms/queue/iris/Secs - a queue for sending request messages (see Section 5.1.39, "SecsApiRequest")
- jms/topic/iris/Secs/client - a queue for receiving messages by subscribers with a unique client identifier (see Section 5.1.38, "SecsApiReply").
- jms/topic/iris/Secs/broadcast - a queue for receiving broadcast messages (not working as of 20.06.2020).

Currently, it is possible to execute the following requests:

- request to receive an instrument list (see Section 5.7.6, "InstrumentsRequest")
- request to receive instrument and security attributes (see Section 5.7.2, "InstrAttrsRequest")
- search request to receive an instrument list (see Section 5.7.8, "SearchInstrumentsRequest");

When executing request to receive an instrument list, a filtering condition can be specified (see Section 5.7.4, "InstrumentsFilter"). Filter can be set to any combination of the following types of securities (see SecType):

- Shares
- Bonds (divided into government and corporate)
- SBS
- Units
- Futures contracts
- ETF
- Depository receipts

Trading platform, required record statuses (see Section 5.7.12, “SecurityRecordStatus”), attribute of belonging to government securities (see Section 5.7.10, “GovernmentSelector”), list of primary keys of securities and instruments records in the IRIS IS and list of codes of securities and instruments can be specified. Not only codes but also code templates can be transmitted for filtering (see Section 3.22, “Filtering Templates”).

Note on formation of the condition when filling in the `sec_type` and `government` fields in the filter: if the `sec_type field = STOCKS, BONDS, GDRs`, and `government = YES`, then a condition of the form "`sec_type = STOCKS OR (sec_type = BONDS AND government = YES) OR sec_type = GDRs`" will be formed.

In the case of search by ISIN for older securities, the search is also performed by NIN and other possible identifiers.

Filtering can also be performed given versioning of records in the IRIS storage. This makes it possible to organize a mode of working with IRIS API in which it is possible to download only new or changed records of instruments and securities (as of 20.06.2020, it does not work).

The request can contain an instruction to limit the returned result to the required range (see Section 5.2.17, "Range") or the size of the received record package.

3.21. Indicators

When working with IRIS API in terms of stock market indicators, three message queues are used:

- jms/queue/iris/Indicators - a queue for sending request messages (see Section 5.1.19, "IndicatorsApiRequest")
- jms/topic/iris/Indicators/client - a queue for receiving messages by subscribers with a unique client identifier (see Section 5.1.18, "IndicatorsApiReply").
- jms/topic/iris/Indicators/broadcast - a queue for receiving broadcast messages (see Section 5.18.18, "IndicatorsUpdate")

There are four basic request types:

- request for a count of records matching the filter condition (see Section 5.18.10, "IndicatorsCountRequest")
- request for a list of indicator values that match the filter condition (see Section 5.18.16, "IndicatorsRequest").
- request for information about plotting the graphs for indicators (see Section 5.18.4, "IndGraphRequest").
- request for a list of the latest indicator values (see Section 5.18.6, "IndQuotesRequest").
- request for information about changes in the value of a selected indicator for a year before a specified date (see Section 5.18.2, "IndChangeRequest").

When executing a request, filtering conditions (see Section 5.18.13,
"IndicatorsFilter") and sorting conditions (see Section 5.18.20, "IndicatorsSortField") can be specified.

Filtering can be performed given versioning of records in the IRIS storage. This makes it possible to organize a mode of working with IRIS API in which it is possible to download only new or changed indicator records.

The request can contain an indication to limit the returned result to the required range (see Section 5.2.17, "Range") or the size of the received record package.

IndicatorsUpdate broadcast messages do not contain indicator codes by definition, since it is anonymized information. To obtain full information about indicators, in reply to the IndicatorsUpdate message, it is necessary to send a request on your behalf for a list of indicator values with a filter from record codes and receive full information in reply to the "jms.topic.iris.Indicators.client" queue. Full information will only be received if it is available to the user.

3.22. Filtering Templates

When transmitting requests containing filters (see, for example, Section 5.7.4, "InstrumentsFilter"), template expressions can be used.

To search for several characters at once, the "*" template is used in the expression.

When searching for several unknown characters, the "*" template will be replaced with either an empty string or a string consisting of at least one character. For example, to search for the codes "KZTK", "KZTO" or "KZTKb1", you can use the following template expression:

```
KZT*
```

The "*" template can also be used inside an expression. For example, "MUM084_0001" or "MUM072_0001":

```
MUM*_0001
```

The "*" template can be included in an expression more than once. As a result, to search for the codes "USDKZT_0_01M", "USDKZT_0_03M", "USDKZT_1_01M" or "USDKZT_1_03M", you can use an expression of this type:

```
USD*_*_*M
```

The possibility of using templates is specified in the description of a specific field.

4. Reference tables

Below are reference tables with codes used when transmitting messages within IRIS API.

Table 1. News message column

Code	Parent code	Description
1		News
4	1	Policy
5	1	Economy
6	1	Markets
7	1	Laws
8	1	Interviews
9	1	Reviews
22	1	Business
2		Events
3		Analystics
10		Main
11	10	Main on the site
12	10	Main in the column
15		Articles
16		Glossary
18		Media
17	18	Viewo news
19		Fee availability
14	19	Free news
24	19	Fee-based news

20		Sources
13	20	KASE news
25	13	Exchange
26	13	Companies
32	13	Markets
21		Statistical content
23		NBFR
27		Sliders
28	27	Slider (mobile application)
29		Polls
30	29	Polls (mobile application)
31	29	Polls (exchange simulator)
33	20	KASE notices
34	33	Notice ssections
36	34	Markets
Code	**Parent code**	**Description**
37	36	Shares
38	36	Bonds
39	36	Currency
40	36	GS
41	34	Corporate events
42	34	Announcements of partners
43	34	Events
46	34	Other
47	33	Important notices

Table 2. File types

Code	Ext	MIME	Type name
1	txt	text/plain	Text file
2	odt	application/vnd.oasis.opendocument.text	OpenOffice Writer
3	ods	application/vnd.oasis.opendocument.spreadsheet	OpenOffice Calc
4	doc	application/msword	Microsoft Word
5	xls	application/vnd.ms-excel	Microsoft Excel
6	pdf	application/pdf	Acrobat Reader

7	html	text/html	HTML
8	png	image/png	PNG image
9	jpg	image/jpeg	JPEG image
10	gif	image/gif	GIF animation
11	zip	application/zip	ZIP-archive
12	ppt	application/vnd.ms-powerpoint	PPT presentation
13	swf	application/x-shockwave-flash	Adobe Flash application
14	tif	image/tif	PNG image

Table 3. Codes of deals statuses

Code	Name
1	Calculated
2	Rejected by user of the Confirmation System
3	Rejected by depositary
4	Pending confirmation of a user of the confirmation system
5	Pending settlements in depository
6	Rejected by user of the Confirmation System on the part of the counterparty
7	Pending confirmation by user of the Confirmation System on the part of the counterparty
8	Pending confirmation by buyer
9	Paid by buyer
10	Unpaid
11	Securities delivered
12	Not delivered
13	Pending change
14	Pending approval
15	Rejected by depositary
16	Rejected by depositary due to lack of money
17	Rejected by depositary due to lack of securities
18	Pending consent of counterparty to make repeated settlements under deal due to lack of money
19	Pending consent of counterparty to conduct repeated settlements under deal (lack of securities)
20	Refusal to make repeated settlements due to lack of money
21	Refusal to make repeated settlements due to lack of securities

22	Refusal of money seller of make repeated settlements after replacement of an "automatic" repo object
23	Cancelled
24	Rejected deal (no timely confirmation of custody)
25	Rejected by buyer
26	Rejected by seller
27	Rejected by parties
28	Expects settlements
29	Liquidation
30	Pending cancellation T+2
31	Cancelled T+2
32	Pending clearing (settlements)
33	Default
34	Not satisfied, not calculated
35	Changed
37	satisfied
38	withdrawn

Table 4. Currency Sessions Codes

Code	Session
1	Morning
2	Day
3	Evening
4	End-to-end
5	Additional

Table 5. Currency codes

Code	Name	Description
1	KZT	Kazakhstan tenge
2	USD	US dollar
3	EUR	Euro
4	RUB	Russian ruble
5	AUD	Australian dollar
6	GBP	English pound
7	BYR	Belarusian ruble

8	DKK	Danish crown
9	AED	UAE dirham
10	CAD	Canadian dollar
11	CNY	Chinese yuan
12	KWD	Kuwaiti dinar
13	KGS	Kyrgyz som
14	LVL	Latvian lat
15	LTL	Lithuanian lita
16	MDL	Moldavian leu
17	NOK	Norwegian krone
18	PLN	Polish zloty
19	SAR	Saudi Arab riyal
20	XDR	Special Drawing Rights
21	SGD	Singapore dollar
22	TRY	Turkish lira
23	UZS	Uzbekistani sum
24	UAH	Ukrainian hryvnia
25	SEK	Swedish krona
26	CHF	Swiss franc
27	EEK	Estonian krone
28	ZAR	South-African rand
29	KRW	South Korean won
30	JPY	Japanese yen
31	HUF	Hungarian forint
32	CZK	Czech koruna
33	TJS	Tajikistani somoni
34	BRL	Brazilian real
35	MYR	Malaysian ringgit
36	HKD	Hong Kong dollar
37	AZN	Azerbaijani manat
38	INR	Indian rupee
39	THB	Thai baht
40	AMD	Armenian dram
41	GEL	Georgian lari

42	IRR	Iranian rial
43	MXN	Mexican peso
44	BYN	Belarusian ruble

Table 6. Codes of currency instruments

Code	**Instrument**	**Currency**	**Description**
1	USDKZT_TOD	USD	US dollar, traded with T+0 settlement (with execution of deals on the trading day)
2	USDKZT_TOM	USD	US dollar, traded with T+1 settlement (with execution of deals on the next business day after the trading day)
3	USDKZT_SPT	USD	US dollar, traded with T+2 settlement
4	USDKZT_0_001	USD	Currency swap, opening deal of which is a deal with USDKZT_TOD, and closing deal is USDKZT_TOM
5	USDKZT_1_002	USD	Currency swap, opening deal of which is a trade with USDKZT_TOM, and closing deal is USDKZT_SPT
6	USDKZT_0_002	USD	Currency swap, opening deal of which is a deal with USDKZT_TOD, and closing deal is USDKZT_SPT
7	EURKZT_TOD	EUR	Euro, traded with T+0 settlement (with execution of deals on the trading day)
8	EURKZT_TOM	EUR	Euro, traded with T+1 settlement (with execution of deals on the next business day after the trading day)
9	RUBKZT_TOD	RUB	Russian ruble, traded with T+0 settlements (with execution of deals on the day of trading)
10	RUB_TOM	RUB	Russian ruble, traded with T+1 settlements (with execution of deals on the next business day after the day of trading)
11	USD	USD	US dollar, traded with T+0 settlements (with execution of deals on the day of trading)
12	USD_SPOT	USD	US dollar, traded with T+2 settlements
14	EUR	EUR	Euro, traded with T+0 settlements (with execution of deals on the day of trading)
15	RUR_TOM	RUB	Russian ruble, traded with T+1 settlements (with execution of deals on the day of trading)
16	RUB	RUB	Russian ruble, traded with T+0 settlements (with execution of deals on the trading day)
17	RUR	RUB	Russian ruble, traded with T+0 settlements (with execution of deals on the trading day)
20	SOM	KGS	
21	UKR	UAH	

Code	Instrument	Currency	Description
22	EURUSD_TOD	EUR	Euro, traded with T+0 settlements (with execution of deals on the trading day)
23	EURKZT_0_001	EUR	Currency swap, opening deal of which is a deal with EURKZT_TOD, and closing deal is EURKZT_TOM
24	EURKZT_1_002	EUR	Currency swap, opening deal of which is a deal with EURKZT_TOM, and closing deal is EURKZT_SPT
25	EURKZT_0_002	EUR	Currency swap, opening deal of which is a deal with EURKZT_TOD, and closing deal is EURKZT_SPT
26	EURUSD_TOM	EUR	Euro traded with settlements T+1 (with execution of deals on the next business day after the trading day)
27	EURUSD_SPT	EUR	Euro traded with settlements T+2
28	EURKZT_SPT	EUR	Euro traded with settlements T+2
29	CNYKZT_TOD	CNY	Chinese Yuan with settlements in tenge
30	CNYKZT_TOM	CNY	Chinese Yuan with settlements in tenge on the day T+1
31	RUBKZT_TOM	RUB	Russian ruble, settlements on the day T+1
32	CNYKZT_SPT	CNY	Chinese Yuan with settlements T+2
33	CNYKZT_0_001	CNY	Currency swap, opening deal of which is a deal with CNYKZT_TOD, and closing deal is CNYKZT_TOM
34	CNYKZT_0_002	CNY	Currency swap, opening deal of which is a deal with CNYKZT_TOD, and closing deal is CNYKZT_SPT
35	CNYKZT_1_002	CNY	Currency swap swap, opening deal of which is a deal with CNYKZT_TOM, and closing deal is CNYKZT_SPT
36	RUBKZT_SPT	RUB	Russian ruble with settlements T+2
37	RUBKZT_0_001	RUB	Currency swap, opening deal of which is a deal with RUBKZT_TOD, and closing deal is RUBKZT_TOM
38	RUBKZT_0_002	RUB	Currency swap, opening deal of which is a deal with RUBKZT_TOD, and closing deal is RUBKZT_SPT
39	RUBKZT_1_002	RUB	Currency swap, opening deal of which is a deal with RUBKZT_TOM, and closing deal is RUBKZT_SPT
40	USDKZT_01M	USD	US dollar with settlements in tenge for 1 month

41	USDKZT_01W	USD	US dollar with settlements in tenge on day T+7
42	USDKZT_03M	USD	US dollar with settlements in tenge for 3 months
43	USDKZT_0_01M	USD	Currency swap deal
44	USDKZT_0_01W	USD	Currency swap deal
45	USDKZT_0_03M	USD	Currency swap deal
46	USDKZT_1_01M	USD	Currency swap deal
47	USDKZT_1_01W	USD	Currency swap deal
48	USDKZT_1_03M	USD	Currency swap deal
49	USDKZT_06M	USD	US dollar settled in tenge for 6 months
50	USDKZT_01Y	USD	US dollar settled in tenge for 1 year
51	USDKZT_0_06M	USD	USD_TOD_6Month
52	USDKZT_0_01Y	USD	USD_TOD_1Year
Code	**Instrument**	**Currency**	**Description**
53	USDKZT_1_01Y	USD	USD_TOM_1Year
54	USDKZT_1_06M	USD	USD_TOM_6Month
59	USDCNY_TOD	USD	USD with settlements in CNY, T+0
60	USDCNY_TOM	USD	USD with settlements in CNY, T+1
61	USDCNY_SPT	USD	USD with settlements in CNY, T+2
62	EURCNY_TOD	EUR	EUR with settlements in CNY, T+0
63	EURCNY_TOM	EUR	EUR with settlements in CNY, T+1
64	EURCNY_SPT	EUR	EUR with settlements in CNY, T+2
65	USDRUB_TOD	USD	USD with settlements in RUB, T+0
66	USDRUB_TOM	USD	USD with settlements in RUB, T+1
67	USDRUB_SPT	USD	USD with settlements in RUB, T+2

Table 7. Codes of indicator types

Code	Description	Unit	Period
KASE shares			
2	KASE index		after deal
7	Volume of deals	k USD	once a day
8	Capitalization	mln USD	once a day
Corporate bonds			
10	KASE_BP index		once a day
11	Capitalization	mln USD	once a day
12	Volume of deals	k USD	once a day

Code	Description	Unit	Period
34	Capitalization of eurobonds	mln USD	once a day
9	KASE_BY		once a day
99	KASE_BC index		settlement suspended
100	KASE_BMC index		once a day
101	KASE_BMY indicator		once a day
Government securities			
109	Index of government bonds of RK Ys	% p.a.	once a day
110	Index of government bonds of RK CPs	% p.a.	once a day
111	Index of government bonds of RK DPs	% p.a.	once a day
112	Index of government bonds of RK Ym	% p.a.	once a day
113	Index of government bonds of RK CPm	% p.a.	once a day
114	Index of government bonds of RK DPm	% p.a.	once a day
115	Index of government bonds of RK Yl	% p.a.	once a day
116	Index of government bonds of RK CPl	% p.a.	once a day
117	Index of government bonds of RK DPl	% p.a.	once a day
124	Price index of "net prices" of government securities without segmentation term (KZGB_CP). Calculated from 02.10.2023.	KZT	once a day
125	Price index of aggregate income of GS without segmentation term (KZGB_DP). Calculated from 02.10.2023.	KZT	once a day
126	Indicator of GS yield parameter without segmentation term (KZGB_Y). Calculated from 02.10.2023.	% p.a.	once a day
127	Index of aggregate inform for GS - DTM 31 and more KZGB_DPm1m). Calculated from 01.04.2022.	KZT	once a day
128	Index of net prices for GS - DTM 31 days and more (KZGB_CPm1m). Calculated from 01.04.2022.	KZT	once a day
129	Yield index for GS - DTM 31 days and more (KZGB_Ym1m). Calculated from 01.04.2022.	%	once a day
130	Average weighted duration of index with maturity of 31 days and more (Duration). Calculated from 01.04.2022.	years	once a day
131	Modified average weighted duration of index with maturity of 31 days and more (ModDuration). Calculated from 01.04.2022.	years	once a day

Foreign currencies			
14	USDKZT_TOD		once a day
15	Volume USDKZT_TOD	mln USD	once a day
16	EURKZT_TOD		once a day
17	EURUSD_TOD		once a day
18	RUBKZT_TOD		once a day
19	USDKZT_TOM		once a day
20	Volume USDKZT_TOM	mln USD	once a day
102	USDKZT_DAY		3 times a day
Money market			
119	MM Index	% p.a.	once a day
120	SWAP-1D	% p.a.	after deal
121	SWAP-2D	% p.a.	after deal
Repo			
21	TONIA	% p.a.	once a day
22	TWINA	% p.a.	after deal
23	1 day (KZT)	% p.a.	once a day
24	2-3 days (KZT)	% p.a.	once a day
25	7-10 days (KZT)	% p.a.	once a day
26	11-14 days (KZT)	% p.a.	once a day
27	Volume of trades	mln KZT	once a day
103	TRION	% p.a.	after deal
104	TONIA Compounded Index	% p.a.	once a day
105	TONIA Compounded Rate_1M	% p.a.	once a day
106	TONIA Compounded Rate_3M	% p.a.	once a day
107	TONIA Compounded Rate_6M	% p.a.	once a day
Code	**Description**	**Unit**	**Period**
Inter-bank deposits			
28	KIBOR-1W (KZT)	% p.a.	once a day
29	KIBOR-2W (KZT)	% p.a.	once a day
30	KIBOR-1M (KZT)	% p.a.	once a day
31	KIBOR-2M (KZT)	% p.a.	once a day
32	KIBOR-3M (KZT)	% p.a.	once a day

33	KazPrime-3M (KZT)	% p.a.	once a day
94	KIMEAN-1W (KZT)	% p.a.	once a day
95	KIMEAN-2W (KZT)	% p.a.	once a day
96	KIMEAN-1M (KZT)	% p.a.	once a day
97	KIMEAN-2M (KZT)	% p.a.	once a day
98	KIMEAN-3M (KZT)	% p.a.	once a day
Macroeconomic parameters of RK			
122	Annual inflation	y/y%	once a month
123	Base rate	% p.a.	once a month

Table 8. Identifier of field which was subject to check upon user registration

Code	Description
fullname	Surname, given name and patronymic name
email	Electronic address
username	User name
password	Password
tel	Telephone
address	Address
captcha	Robot check field

Table 9. Information products

Code	Name
10	News
20	Issuers
21	Securities
30	Deals (Corporater shares)
32	Deals (Corporater bonds)
33	Deals (GS)
34	Deals (REPO)
35	Deals (Currency)
40	Orders (Corporater shares)
41	Orders (Corporater bonds)
42	Orders (GS)
44	Orders (Currency)
50	Trading results
80	Currency rates (exchange)

81	Currency rates (NBK)
90	Indicators

4.1. Structure of the JSON file with issuer attributes

Below is a description of the structure of the JSON file with issuer attributes, which is returned in reply to the IssuerAttrsRequest request if the JSONAttrsParams type is passed in the parameters.

The file contains the following main sections:

- Issuer attributes;
- General information;
- Representations;
- Licenses;
- Company ratings;
  - Shareholders;
  - Financial indicators;

4.1.1. Main sections of the file

```
{
   "main": { ❶
...
   },
   "general": { ❷
...
   },
   "representations": { ❸
...
   },
   "licenses": { ❹
...
   },
   "ratings": { ❺
...
   },
   "shareholders": { ❻
...
   },
   "indicators": { ❼
...
   }
}
```


❶ Issuer attributes;
❷ General information;
❸ Representations;
❹ Licenses;
❺ Company ratings;
❻ Shareholders;
❼ Financial indicators;

4.1.2. “Issuer Attributes” Section

```
"main": {
   "code": "KZTK", ❶
   "short_name": "JSC \"Kazakhtelecom\"", ❷
   "full_name": "Joint Stock Company \"Kazakhtelecom\"", ❸
   "tiny_name": "Kazakhtelecom", ❹
   "actual_address": "Astana, Yesil district, Sauran str., 12", ❺
   "legal_address": "Astana, Yesil district, Sauran str., 12", ❻
   "phones": "(7172) 59 16 00, 58 08 39; (727) 258 72 19, 258 72 15", ❼
   "fax": "(7172) 58 77 24", ❽
   "site": "http://www.telecom.kz/", ❾
   "email": "telecom@telecom.kz, ir@telecom.kz", ❿
   "logo": "string", ⓫
   "bin": "941240000193", ⓬
   "legal_registration": "1994-12-01", ⓭
   "contacts": "string" ⓮
   "keywords": "Kazakhtelecom" ⓯
   "stopwords": "string" ⓰
}
```

❶ Issuer code;
❷ Short name;
❸ Full name;
❹ Short name;
❺ Visiting address;
❻ Registered office;
❼ Telephone;
❽ Fax;
❾ Representation in Internet;
❿ Email;
⓫ Logo;
⓬ BIN;
⓭ Date of original registration as a legal entity;
⓮ Contacts;
⓯ Key words;
⓰ Does not contain key words;

4.1.3. “General Information” Section

```
"general": {
   "general_info": {
      "economic_sector": "50 Provision of telecommunication services", ❶
      "primary_activity": "telephone, telegraph, television and radio communication
        of all types
      "parent_company": "string", ❸
      "initial_registration": "string", ❹
      "last_reregistration": "string", ❺
      "rnn": "string", ❻
      "position": "Chairman of the Management Board", ❼
      "full_name": "Yesekeyev Kuanyshbek Bakhytbekovich", ❽
      "publications": "Kazakhstanskaya Pravda, Yegemen Kazakhstan" ❾
   }
}
```

❶ Economic sector;
❷ Core activities;
❸ Holding company’s name;
❹ Date of original registration (line of YYYY-MM-DD format);
❺ Date of recent state re-registration (line of YYYY-MM-DD format);
❻ TRN;
❼ Position of chief executive;
❽ Full name of of chief executive;
❾ Printed publications

4.1.4. “Representations” Section

```
"representations": {
   "representations_table": [
      {
         "representation_name": "joint stock company \"Kazakhtelecom\"" ❶
      },
      {
         "representation_name": "JSC \"First Heartland Securities\""
      }
   ]
}
```

❶ Representation name;

4.1.5. “Licenses” Section

```
"licenses": {
   "licenses_table": [
      {
         "license_name": "string" ❶
      }
   ]
}
```

❶ License name;

4.1.6. “Company Ratings” Section

The table contains current ratings of the company on a date of execution the request.

```
"ratings": {
   "ratings_table": [
      {
         "date": "2021-12-10", ❶
         "rating_agency": "Fitch Ratings", ❷
         "rating_type": "Long-term national scale rating", ❸
         "rating": "AA+(kaz)", ❹
         "rating_forecast": "stable" ❺
      }
   ]
}
```

❶ Rating assignment date (line of YYYY-MM-DD f ormat);
❷ Rating agency;
❸ Rating type;
❹ Rating;
❺ Forecast;

4.1.7. “Shareholders” Section

```
"shareholders": {
   "shareholders_table": [
      {
          "holder_name": "JSC First Heartland Jusan Bank", ❶
          "shares_qnt": 983350, ❷
          "shares_prop": 9.18501583781243, ❸
          "privileged": 1833000, ❹
          "all_shares_qnt": 983350, ❺
          "all_shares_prop": 8.10239896431673 ❻
      }
   ],
   "shareholders_summary": { ❼
      "common": 10922876, ❽
      "privileged": 1213653, ❾
      "common_placed": 10922876, ❿
      "privileged_placed": 1213653, ⓫
      "common_repurchased": 216852, ⓬
      "privileged_repurchased": 914868 ⓭
   }
   }
```

❶ Holder’s name;
❷ Number of ordinary shares;
❸ Share of ordinary shares;
❹ Preference shares;
❺ Number of shares (total);
❻ Number of shares (total);
❼ Aggregate data of shareholders;
❽ Number of authorized ordinary shares;
❾ Number of authorized preference shares;
❿ Number of placed ordinary shares;
⓫ Number of placed preference shares;
⓬ Number of redeemed ordinary shares;
⓭ Number of redeemed preference shares;

4.1.8. “Financial Indicators” Section

The table contains financial indicators for the last four reporting periods (if any).

```
"indicators": {
   "indicators_table": [
      {
          "date": "2023-07-01", ❶
          "data": {
              "authorized_capital": 5070915000, ❷
              "own_capital": 791406287000, ❸
              "total_assets": 1327342046000, ❹
              "net_income": 58519195000, ❺
              "book_value": 10000.00, ❻
              "currency": "KZT", ❼
              "roa": 4.41, ❽
              "roe": 7.39 ❾
          }
      }
   ]
}
```

❶ Date of reporting period;
❷ Authorized capital;
❸ Own capital;
❹ Total assets;
❺ Net income;
❻ Book value of share;
❼ Currency;
❽ ROA;
❾ ROE;

5. IRIS API Messages

This section describes all messages transmitted within IRIS API. Methods for using the messages were described earlier in this document (see Section 3, “Use of IRIS API”).

Messages are divided into the following main groups:

- **Main messages** - the main IRIS API messages
- **Basic types** - basic types (date, time, decimal, etc.)
- **Open information** - messages to transmit open information (see Section 3.1, "Open Information")
- **General information** - messages to transmit general information (see Section 3.2, "General Information")
- **Stock market deals** - messages to transmit information about deals with securities (see Section 3.4, "Stock Market Deals")
- **Stock market orders** - messages to transmit information about orders to buy securities (see Section 3.5, "Stock Market Orders")
- **Stock market order book** - messages to transmit information about the stock market order book (see Section 3.7, "Stock Market Order Book")
- **Stock market graphs** - messages for receiving data for constructing graphs of instrument prices and trading volumes (see Section 3.8, "Stock Market Graphs")

- **Currency market deals** - messages to transmit information about currency market deals (see Section 3.9, "Currency Market Deals")
- **Currency market orders** - messages to transmit information about currency market orders (see Section 3.10, "Currency Market Orders")
- **Currency market order book** - messages to transmit information about the currency market order book (see Section 3.12, "Currency Market Order Book")
- **Currency market graphs** - messages for receiving data for constructing price and trading volume graphs on the currency market (see Section 3.13, "Currency Market Graphs")
- **Repo market deals** - messages to transmit information about repo market deals (see Section 3.14, "Repo Market Deals")
- **Repo market orders** - messages to transmit information about currency market orders (see Section 3.15, "Repo Market Orders")
- **Repo market order book** - messages to transmit information about order book in the repo market (see Section 3.16, “Order Book in the Repo Market”)
- **Issuer attributes** - messages to transmit information about securities (see Section 3.19, “Issuer Attributes”)
- **Securities attributes** - messages to transmit information about securities (see Section 3.20, “Securities and Instrument Attributes”)
- **Securities trading results** - messages to transmit information about securities trading, grouped by period (see Section 3.6, “Stock Market Trading Results”)
- **Currency trading results** - messages to transmit information about trading on the currency market, grouped by period (see Section 3.11, “Currency Trading Results”)
- **User watchlist** - messages for work with the user watchlist (see Section 3.17, “User Watchlist”)
- **News** - messages to transmit information about news (see Section 3.18, “News Messages”)
- **Indicators** - messages to transmit information about stock market indicators (see Section 3.21, "Indicators")

5.1. Main messages

Description of all messages transmitted within IRIS API.

5.1.1. ActualListsApiReply

This message is outdated starting from API version 2.8.0.

IRIS API reply with information about the current state of the lists of instruments and indicators recommended for display to the end user.

The messages actual_lists_reply and error_message are mutually exclusive.

To receive this message, it is necessary to subscribe to the topic: "jms.topic.iris.Info.client" (see Section 3.2, "General Information").

Table 10. `ActualListsApiReply` message fields

Name	Type	Description
serial_num	int64	Serial number of the request to which this reply was received.
error_message	ErrorMessage	Reply with an error message. Mutually exclusive with actual_lists_reply.
actual_lists_reply	ActualListsReply	Reply with a count of records. Mutually exclusive with error_message.

5.1.2. ActualListsApiRequest

This message is outdated starting from API 2.8.0.

IRIS API request to receive information about the current state of the lists of instruments and indicators recommended for display to the end user.

Sent to the queue: "jms.queue.iris.Info" (see Section 3.2, "General Information").

Table 11. `ActualListsApiRequest` message fields

Name	Type	Description
serial_num	int64	Serial number of the request being sent.

5.1.3. ActualListsReply

This message is outdated starting from API 2.8.0.

Message with information about the current state of the lists of instruments and indicators recommended for display to the end user.

Table 12. `ActualListsReply` message fields

Name	Type	Description
main_shares_ids	int32 (array)	List of shares recommended for display to the end user first. The field contains primary keys of the instrument records in IRIS IS.
main_cur_ids	int32 (array)	List of currency instruments recommended for display to the end user first. The field contains primary keys of the currency instrument records in IRIS IS.

actual_cur_ids	int32 (array)	List of current currency instruments that can be displayed to the end user. The field contains primary keys of the currency instrument records in IRIS IS.
Name	**Type**	**Description**
indicator_groups	IndicatorsGroup (array)	List of indicators recommended for display to the end user, divided into groups.
hide_empty_sections	bool	Parameter that determines whether empty sections should be hidden from the user (true - hide, false - not hide).

5.1.4. CurDealsApiReply

IRIS API reply on currency market deals.

The deals_count_reply, deals_reply and error_message messages are mutually exclusive.

To receive this message, you must subscribe to the topic:

"jms.topic.iris.CurDeals.client" (see Section 3.9, "Currency Market Deals").

Table 13. `CurDealsApiReply` message fields

Name	Type	Description
serial_num	int64	Serial number of the request to which this reply was received.
error_message	ErrorMessage	Reply with an error message. Mutually exclusive with deals_count_reply, deals_reply.
deals_count_reply	CurDealsCountReply	Reply with a count of deals. Mutually exclusive with deals_reply, error_message.
deals_reply	CurDealsReply	Reply with a count of deals. Mutually exclusive with deals_count_reply, error_message.
export_reply	CurDealsExportReply	Reply with a count of deals.

5.1.5. CurDealsApiRequest

IRIS API request for deals on the currency market.

The count_request and deals_request messages are mutually exclusive.

Sent to the queue: "jms.queue.iris.CurDeals" (see Section 3.9, "Currency Market Deals").

Table 14. `CurDealsApiRequest` message fields

Name	Type	Description

serial_num	int64	Serial number of the request being sent.
count_request	CurDealsCountRequest	Request for a count of deals on the currency market. Mutually exclusive with deals_request.
deals_request	CurDealsRequest	Request for a list of deals on the currency market.
Name	**Type**	**Description**
		Mutually exclusive with count_request.
export_request	CurDealsExportRequest	Request for information about currency market deals as a file in the specified format.

5.1.6. CurGraphApiReply

IRIS API reply with information for plotting graphs on the currency market.

To receive this message, you must subscribe to the topic:

"jms.topic.iris.CurGraph.client" (see Section 3.13, "Currency Market Graphs").

Table 15. `CurGraphApiReply` message fields

Name	Type	Description
serial_num	int64	Serial number of the request to which this reply was received.
error_message	ErrorMessage	Reply with an error message.
graph_reply	CurGraphReply	Information to plot currency market graphs.

5.1.7. CurGraphApiRequest

IRIS API request to receive data for plotting currency market graphs.

Sent to the queue: "jms.queue.iris.CurGraph" (see Section 3.13, "Currency Market Graphs").

Table 16. `CurGraphApiRequest` message fields

Name	Type	Description
serial_num	int64	Serial number of the request being sent.
request	CurGraphRequest	Request for information to plot currency market graphs.

5.1.8. CurOrderbookApiReply

IRIS API reply for the currency market order book.

The orderbook_reply and error_message messages are mutually exclusive.

To receive this message, it is necessary to subscribe to the topic: "jms.topic.iris.CurOrderbook.client" (see Section 3.12, "Currency Market Order Book").

Table 17. `CurOrderbookApiReply` message fields

Name	Type	Description
serial_num	int64	Serial number of the request to which this reply was received.
error_message	ErrorMessage	Reply with an error message. Mutually exclusive with orderbook_reply.
orderbook_reply	CurOrderbookReply	Reply with a currency market order book. Mutually exclusive with error_message.

5.1.9. CurOrderbookApiRequest

IRIS API request for a currency market order book.

Sent to the queue: “jms/topic/iris/CurOrderbook” (see Section 3.12, “Currency Market Order Book”).

Table 18. `CurOrderbookApiRequest` message fields

Name	Type	Description
serial_num	int64	Serial number of the request being sent.
request	CurOrderbookRequest	Request for information about the order book on the currency market.

5.1.10. CurOrdersApiReply

IRIS API request for currency market orders.

The orders_count_reply, orders_reply, export_reply and error_message messages are mutually exclusive.

To receive this message, it is necessary to subscribe to the topic: "jms.topic.iris.CurOrders.client" (see Section 3.10, “Currency Market Orders”).

Table 19. `CurOrdersApiReply` message fields

Name	Type	Description
serial_num	int64	Serial number of the request to which this reply was received.
error_message	ErrorMessage	Reply with an error message. Mutually exclusive with orders_count_reply, orders_reply, export_reply.
orders_count_reply	CurOrdersCountReply	Reply with a count of orders. Mutually exclusive with orders_reply, error_message.
orders_reply	CurOrdersReply	Reply with a list of orders. Mutually exclusive with orders_count_reply, error_message.

export_reply	CurOrdersExportReply	Reply with a list of orders.

5.1.11. CurOrdersApiRequest

IRIS API request for currency market orders.

The count_request, orders_request and export_request messages are mutually exclusive.

Sent to the queue: "jms.queue.iris.CurOrders" (see Section 3.10, “Currency Market Orders”).

Table 20. `CurOrdersApiRequest` message fields

Name	Type	Description
serial_num	int64	Serial number of the request being sent.
count_request	CurOrdersCountRequest	Request for a count of currency market orders.
Name	**Type**	**Description**
		Mutually exclusive with orders_request.
orders_request	CurOrdersRequest	Request for a list of orders. Mutually exclusive with count_request.
export_request	CurOrdersExportRequest	Request for information about on currency market orders as a file in the specified format.

5.1.12. CurTotalsApiReply

IRIS API reply with information about currency market trades grouped for a period.

The totals_count_reply, totals_reply, quotations_reply, export_reply and error_message messages are mutually exclusive.

To receive this message, it is necessary to subscribe to the topic: "jms.topic.iris.CurTotals.client" (see Section 3.11, “Currency Market Trading Results”).

Table 21. `CurTotalsApiReply` message fields

Name	Type	Description
serial_num	int64	Serial number of the request to which this reply was received.
error_message	ErrorMessage	Reply with an error message.
totals_count_reply	CurTotalsCountReply	Reply with a count of orders at trades.
totals_reply	CurTotalsReply	Reply with a list of orders about trades.
quotations_reply	CurQuotationsReply	Reply with information about currency market quotations.
export_reply	CurTotalsExportReply	Reply with information about formed export file with information about results of stock market trades.

5.1.13. CurTotalsApiRequest

IRIS API request for information about currency market trades grouped for a period.

The count_request, totals_request, quotations_request and export_request messages are mutually exclusive.

Sent to the queue: "jms.queue.iris.CurTotals" (see Section 3.11, “Results of currency market trades”).

Table 22. `CurTotalsApiRequest` message fields

Name	Type	Description
serial_num	int64	Serial number of the request being sent.
count_request	CurTotalsCountRequest	Request for a count of records about currency market trades.
totals_request	CurTotalsRequest	Request for a list of records about trades
quotations_request	CurQuotationsRequest	Request for information about currency market quotations.
export_request	CurTotalsExportRequest	Request for information about results of currency market trades as a file in the specified format.

5.1.14. DealsApiReply

IRIS API request for security deals.

The deals_count_reply, deals_reply, export_reply and error_message messages are mutually exclusive.

To receive this message, it is necessary to subscribe to the topic: "jms.topic.iris.Deals.client" (see Section 3.4, “Stock Market Graphs”).

Table 23. `DealsApiReply` message fields

Name	Type	Description
serial_num	int64	Serial number of the request to which this reply was received.
error_message	ErrorMessage	Reply with an error message.
deals_count_reply	DealsCountReply	Reply with a count of deals.
deals_reply	DealsReply	Reply with a count of deals.
export_reply	DealsExportReply	Reply with a count of deals.

5.1.15. DealsApiRequest

IRIS API request for security deals.

The count_request, deals_request and export_request messages are mutually exclusive.

Sent to the queue: "jms.queue.iris.Deals" (see Section 3.4, “Stock Market Graphs”).

Table 24. `DealsApiRequest` message fields

Name	Type	Description
serial_num	int64	Serial number of the request being sent.

count_request	DealsCountRequest	Request for a count of deals.
deals_request	DealsRequest	Request for a list of deals.
export_request	DealsExportRequest	Request for information about deals as a file in the specified format.

5.1.16. GraphApiReply

IRIS API reply with information to plot stock market graphs.

To receive this message, it is necessary to subscribe to the topic: "jms.topic.iris.Graph.client" (see Section 3.8, “Stock Market Graphs”).

The error_message, graph_reply and heatmap_reply messages are mutually exclusive.

Table 25. `GraphApiReply` message fields

Name	Type	Description
serial_num	int64	Serial number of the request to which this reply was received.
error_message	ErrorMessage	Reply with an error message.
graph_reply	GraphReply	Information to plot graphs.
Name	**Type**	**Description**
heatmap_reply	HeatmapReply	Reply with information to plot a heat map.

5.1.17. GraphApiRequest

IRIS API request for data to plot graphs.

Sent to the queue: "jms.queue.iris.Graph" (see Section 3.8, “Stock Market Graphs”).

The graph_request and heatmap_request messages are mutually exclusive.

Table 26. `GraphApiRequest` message fields

Name	Type	Description
serial_num	int64	Serial number of the request being sent.
graph_request	GraphRequest	Request for data to plot stock market graphs.
heatmap_request	HeatmapRequest	Request for data to plot heat map.

5.1.18. IndicatorsApiReply

IRIS API reply with a list of values of indicators that match the filtering conditions.

The indicators_count_reply, indicators_reply, graph_reply, quotes_reply, change_reply, export_reply and error_message messages are mutually exclusive.

To receive this message, it is necessary to subscribe to the topic: "jms.topic.iris.Indicators.client" (see Section 3.21, “Indicators”).

Table 27. `IndicatorsApiReply` message fields

Name	Type	Description
serial_num	int64	Serial number of the request to which this reply was received.
error_message	ErrorMessage	Reply with an error message. Mutually exclusive with indicators_count_reply, indicators_reply.
indicators_count_reply	IndicatorsCountReply	Reply with a count of records. Mutually exclusive with indicators_reply, error_message.
indicators_reply	IndicatorsReply	Reply with a list of values of indicators. Mutually exclusive with indicators_count_reply, error_message.
graph_reply	IndGraphReply	Reply with information to plot indicator graphs.
quotes_reply	IndQuotesReply	Reply with a list of recent values of indicators.
change_reply	IndChangeReply	Reply with information about change in value of a selected indicator for a year before a specified data.
Name	Type	Description
export_reply	IndicatorsExportReply	Reply with a list of indicators.

5.1.19. IndicatorsApiRequest

IRIS API request for a list of values of indicators that match the filtering condition.

The count_request, indicators_request, graph_request, quotes_request, change_request and export_request messages are mutually exclusive.

Sent to the queue: "jms.queue.iris.Indicators" (see Section 3.21, “Indicators”).

Table 28. `IndicatorsApiRequest` message fields

Name	Type	Description
serial_num	int64	Serial number of the request being sent.
count_request	IndicatorsCountRequest	Request for a count of deals on the currency market.
indicators_request	IndicatorsRequest	Request for a list of deals on the currency market.
graph_request	IndGraphRequest	Request for information to plot indicator graphs.
quotes_request	IndQuotesRequest	Request for a list of recent values of indicators.

change_request	IndChangeRequest	Request for information about change in value of a selected indicator for a year before a specified data.
export_request	IndicatorsExportRequest	Request for information about a list of indicators as a file in the specified format.

5.1.20. InfoApiReply

IRIS API reply with general information.

The actual_lists_reply, pools_list_reply, pool_questions_reply, pool_answers_reply, change_pwd_reply, error_message, boards_reply, user_info_reply, user_feedback_reply, trades_schedule_reply, trades_status_reply, brokers_list_reply and broker_message_reply messages are mutually exclusive.

To receive this message, it is necessary to subscribe to the topic: "jms.topic.iris.Main.client" (see Section 3.2, “General Information”).

Table 29. `InfoApiReply` message fields

Name	Type	Description
serial_num	int64	Serial number of the request to which this reply was received.
error_message	ErrorMessage	Reply with an error message.
actual_lists_reply	MainListsReply	Reply with information about a current status of lists of instruments and indicators recommended for display image to the end user.
Name	**Type**	**Description**
pools_list_reply	PoolsListReply	Reply with a list of active polls.
pool_questions_reply	PoolQuestionsReply	Reply with questions of selected poll.
pool_answers_reply	PoolAnswersReply	Reply to a message with sent replies to the poll.
change_pwd_reply	ChangePwdReply	Reply with a result of password change.
boards_reply	BoardsReply	Reply with a list of existing MOEX regimes.
user_info_reply	UserInfoReply	Reply with information about an IRIS user.
user_feedback_reply	UserFeedbackReply	Reply to a user message sent to ensure feedback.
trades_schedule_reply	TradesScheduleReply	Reply with information about trades schedule.
trades_status_reply	TradesStatusReply	Reply with information about trades status.
brokers_list_reply	BrokersListReply	Reply with a list of brokers.

broker_message_reply	BrokerMessageReply	Reply to a message with a request sent by user to broker.

5.1.21. InfoApiRequest

IRIS API request to receive general information.

The actual_lists_request, pools_list_request, pool_questions_request, pool_answers_request, change_pwd_request, boards_request, user_info_request, user_feedback_request, trades_schedule_request, trades_status_request, brokers_list_request and broker_message_request messages are mutually exclusive.

Sent to the queue: "jms.queue.iris.Main" (see Section 3.2, “General Information”).

Table 30. `InfoApiRequest` message fields

Name	Type	Description
serial_num	int64	Serial number of the request being sent.
actual_lists_request	MainListsRequest	Request for information about a current state of lists of instruments and indicators recommended for disply to the end user.
pools_list_request	PoolsListRequest	Request for a list of active polls.
pool_questions_request	PoolQuestionsRequest	Request for answers of the selected poll.
pool_answers_request	PoolAnswersRequest	Request for answers to the poll.
change_pwd_request	ChangePwdRequest	Request to change a user password.
boards_request	BoardsRequest	Request for a list of existing MOEX regimes.
user_info_request	UserInfoRequest	Request for information about IRIS user.
user_feedback_request	UserFeedbackRequest	Request to be sent by the user to ensure feedback.
trades_schedule_request	TradesScheduleRequest	Request of information about trades schedule.
Name	**Type**	**Description**
trades_status_request	TradesStatusRequest	Request of information about trades status.
brokers_list_request	BrokersListRequest	Request for a list of brokers.
broker_message_request	BrokerMessageRequest	Request to be sent by the user to broker.

5.1.22. IssuersApiReply

IRIS API reply with information about issuers.

To receive this message, it is necessary to subscribe to the topic: "jms.topic.iris.Issuers.client" (see Section 3.19, “Issuer Attributes”).

Table 31. `IssuersApiReply` message fields

Name	Type	Description
serial_num	int64	Serial number of the request to which this reply was received.
error_message	ErrorMessage	Reply with an error message. Mutually exclusive with issuers_reply, issuer_attrs_reply.
issuers_reply	IssuersReply	Reply with a requested list of instruments. Mutually exclusive with error_message, issuer_attrs_reply.
issuer_attrs_reply	IssuerAttrsReply	Reply with instrument attributes in a requested format. Mutually exclusive with error_message, issuers_reply.

5.1.23. IssuersApiRequest

IRIS API request for information about issuers.

Sent to the queue: "jms.queue.iris.Issuers" (see Section 3.19, “Issuer Attributes”).

Table 32. `IssuersApiRequest` message fields

Name	Type	Description
serial_num	int64	Serial number of the request being sent.
issuers_request	IssuersRequest	Request for a list of issuers. Mutually exclusive with issuer_attrs_request.
issuer_attrs_request	IssuerAttrsRequest	Request to receive issuer atributes. Mutually exclusive with issuers_request.

5.1.24. NewsApiReply

IRIS API reply with information about news messages.
To receive this message, it is necessary to subscribe to the topic: "jms.topic.iris.News.client" (see Section 3.18, “News Messages”).

The error_message, news_count_reply, news_reply, body_reply, refs_reply, files_reply, file_body_reply and full_info_reply messages are mutually exclusive.

Table 33. `NewsApiReply` message fields

Name	Type	Description
serial_num	int64	Serial number of the request to which this reply was received.
error_message	ErrorMessage	Reply with an error message.

news_count_reply	NewsCountReply	Reply with a count of news messages.
news_reply	NewsReply	Reply with a list of titles of news messagges.
body_reply	NewsBodyReply	Reply with selected news messages.
refs_reply	NewsRefsReply	Reply with links for selected news messages.
files_reply	FilesReply	Request to receive a file by its primary key.
file_body_reply	FileBodyReply	Reply with a file or its part as an array of bites.
full_info_reply	FullNewsInfoReply	Reply with full information about news messages
newsusers_add_reply	NewsUsersAddReply	Reply to a request to add news to the list of selected.
newsusers_delete_reply	NewsUsersDeleteReply	Reply to a request to delete news from the list of selected.

5.1.25. NewsApiRequest

IRIS API request for news messages.

Sent to the queue: "jms.queue.iris.News" (see Section 3.18, “News Messages”).

The count_request, news_request, body_request, refs_request, files_request, file_body_request and full_info_request messages are mutually exclusive.

Table 34. `NewsApiRequest` message fields

Name	Type	Description
serial_num	int64	Serial number of the request being sent.
count_request	NewsCountRequest	Request for a number of news messages.
news_request	NewsRequest	Request for a list of titles of news messages.
body_request	NewsBodyRequest	Request to receive selected news m essages.
refs_request	NewsRefsRequest	Request to receive links for selected news messages.
files_request	FilesRequest	Request for attributes of files by their primary keys.
file_body_request	FileBodyRequest	Request for a list of files.
full_info_request	FullNewsInfoRequest	Request to receive full information about news messages.

newsusers_add_request	NewsUsersAddRequest	Request to add news to the list of seslected
newsusers_delete_request	NewsUsersDeleteRequest	Request to remove news from the list of selected

5.1.26. OpenInfoApiReply

Obtaining replyes to IRIS API messages intended to transmit general general information.

The user_reg_reply and error_message messages are mutually exclusive.

To receive this message, it is necessary to subscribe to the topic: "jms.queue.open.iris.Info.client" (see Section 3.1, “Open Informaiton”).

Table 35. `OpenInfoApiReply` message fields

Name	Type	Description
serial_num	int64	Serial number of the request to which this reply was received.
error_message	ErrorMessage	Reply with an error message.
user_reg_reply	UserRegReply	Reply with a result of user registration.
reset_pwd_reply	ResetPwdReply	Reply with a result of password reset.

5.1.27. OpenInfoApiRequest

Sending IRIS API messages intended to transmit general open information.

The user_reg_request and reset_pwd_request messages are mutually exclusive.

Sent to the queue: "jms.queue.open.iris.Info" (see Section 3.1, “Open Information”).

Table 36. `OpenInfoApiRequest` message fields

Name	Type	Description
serial_num	int64	Serial number of the request being sent.
user_reg_request	UserRegRequest	Request for user registration.
reset_pwd_request	ResetPwdRequest	Request for reset of a user password.

5.1.28. OrderbookApiReply

IRIS API request for a stock market order book.

The orderbook_reply anderror_message messages are mutually exclusive.

To receive this message, it is necessary to subscribe to the topic: “jms.topic.iris.Orderbook.client” (see Section 3.7, “Stock Market Order Book”).

Table 37. `OrderbookApiReply` message fields

Name	Type	Description
serial_num	int64	Serial number of the request to which this reply was received.

error_message	ErrorMessage	Reply with an error message. Mutually exclusive with orderbook_reply.
orderbook_reply	OrderbookReply	Reply with a stock market order book. Mutually exclusive with error_message.

5.1.29. OrderbookApiRequest

IRIS API request for a stock market order book.

Sent to the queue: “jms/topic/iris/Orderbook” (see Section 3.7, “Stock Market Order Book”).

Table 38. `OrderbookApiRequest` message fields

Name	Type	Description
serial_num	int64	Serial number of the request being sent.
request	OrderbookRequest	Request for information about a stock market order book.

5.1.30. OrdersApiReply

IRIS API request for orders on the securities market.

The orders_count_reply, orders_reply, export_reply and error_message messages are mutually exclusive.

To receive this message, it is necessary to subscribe to the topic:

"jms.topic.iris.Orders.client" (see Section 3.5, “Stock Market Orders”).

Table 39. `OrdersApiReply` message fields

Name	Type	Description
serial_num	int64	Serial number of the request to which this reply was received.
error_message	ErrorMessage	Reply with an error message. Mutually exclusive with orders_count_reply, orders_reply.
orders_count_reply	OrdersCountReply	Reply with a count of orders. Mutually exclusive with orders_reply, error_message.
orders_reply	OrdersReply	Reply with a list of orders. Mutually exclusive with orders_count_reply, error_message.
export_reply	OrdersExportReply	Reply with a list of orders.

5.1.31. OrdersApiRequest

IRIS API request for orders on the securities market.

The count_request, orders_request and export_request messages are mutually exclusive.

Sent to the queue: "jms.queue.iris.Orders" (see Section 3.5, “Stock Market Orders”).

Table 40. `OrdersApiRequest` message fields

Name	Type	Description
serial_num	int64	Serial number of the request being sent.
count_request	OrdersCountRequest	Request for a count of requests. Mutually exclusive with orders_request.
orders_request	OrdersRequest	Request for a list of orders. Mutually exclusive with count_request.
export_request	OrdersExportRequest	Request for information about orders as a file in the specified format.

5.1.32. RepoApiReply

IRIS API request for Repo Market Deals.

The repo_count_reply, repo_reply, export_reply and error_message messages are mutually exclusive.

To receive this message, it is necessary to subscribe to the topic: "jms.topic.iris.Repo.client" (see Section 3.14, “Repo Market Deals”).

Table 41. `RepoApiReply` message fields

Name	Type	Description
serial_num	int64	Serial number of the request to which this reply was received.
error_message	ErrorMessage	Reply with an error message. Mutually exclusive with repo_count_reply, repo_reply, export_reply.
repo_count_reply	RepoCountReply	Reply with a count of deals on the m arket of repo operations. Mutually exclusive with repo_reply, error_message.
repo_reply	RepoReply	Reply with a count of deals. Mutually exclusive with repo_count_reply, error_message.
export_reply	RepoExportReply	Reply with a count of deals.

5.1.33. RepoApiRequest

IRIS API request for repo market deals.

The count_request, repo_request and export_request messages are mutually exclusive.

Sent to the queue: "jms.queue.iris.Repo" (see Section 3.14, “Repo Market Deals”).

Table 42. `RepoApiRequest` message fields

Name	Type	Description
serial_num	int64	Serial number of the request being sent.
count_request	RepoCountRequest	Request for a count of repo market deals. Mutually exclusive with repo_request.
repo_request	RepoRequest	Request for a list of repo market deals. Mutually exclusive with count_request.
export_request	RepoExportRequest	Request for information about repo market deals as a file in the specified format.

5.1.34. RepoOrderbookApiReply

IRIS API request for a repo market order book.

The orderbook_reply and error_message messages are mutually exclusive.

To receive this message, it is necessary to subscribe to the topic:

«jms.topic.iris.RepoOrderbook.client» (see Section 3.16, “Repo Market Order Book”).

Table 43. `RepoOrderbookApiReply` message fields

Name	Type	Description
serial_num	int64	Serial number of the request to which this reply was received.
error_message	ErrorMessage	Reply with an error message. Mutually exclusive with orderbook_reply.
orderbook_reply	RepoOrderbookReply	Reply with a repo market order book. Mutually exclusive with error_message.

5.1.35. RepoOrderbookApiRequest

IRIS API request for a repo market order book.

Sent to the queue: “jms/topic/iris/RepoOrderbook” (see Section 3.16, “Repo Market Order Book”).

Table 44. `RepoOrderbookApiRequest` message fields

Name	Type	Description
serial_num	int64	Serial number of the request being sent.
request	RepoOrderbookRequest	Request for information about a repo market order book.

5.1.36. RepoOrdersApiReply

IRIS API request for repo requests.

The repo_orders_count_reply, repo_orders_reply and error_message messages are mutually exclusive.

To receive this message, it is necessary to subscribe to the topic: "jms.topic.iris.RepoOrders.client" (see Section 3.15, “Repo Market Orders”).

Table 45. `RepoOrdersApiReply` message fields

Name	Type	Description
serial_num	int64	Serial number of the request to which this reply was received.
error_message	ErrorMessage	Reply with an error message. Mutually exclusive with repo_orders_count_reply, repo_orders_reply.
repo_orders_count_reply	RepoOrdersCountReply	Reply with a count of repo orders. Mutually exclusive with repo_orders_reply, error_message.
repo_orders_reply	RepoOrdersReply	Reply with a list of repo requests. Mutually exclusive with repo_orders_count_reply, error_message.
export_reply	RepoOrdersExportReply	Reply with a list of orders.

5.1.37. RepoOrdersApiRequest

IRIS API request for repo requests.

The count_request and repo_orders_request messages are mutually exclusive.

Sent to the queue: "jms.queue.iris.RepoOrders" (see Section 3.15, “Repo Market Orders”).

Table 46. `RepoOrdersApiRequest` message fields

Name	Type	Description
serial_num	int64	Serial number of the request being sent.
repo_count_request	RepoOrdersCountRequest	Request for a count of repo orders. Mutually exclusive with repo_orders_request.
repo_orders_request	RepoOrdersRequest	Request for a list of repo orders. Mutually exclusive with repo_count_request.
export_request	RepoOrdersExportRequest	Request for information repo orders as a file in the specified format.

5.1.38. SecsApiReply

IRIS API response with information about securities.

The instrument_reply, instr_attrs_reply, search_reply and error_message messages are mutually exclusive.

To receive this message, it is necessary to subscribe to the topic: "jms.topic.iris.Secs.client" (see Section 3.20, “Securities and Instrument Attributes”).

Table 47. `SecsApiReply` message fields

Name	Type	Description
serial_num	int64	Serial number of the request to which this reply was received.
error_message	ErrorMessage	Reply with an error message.
instrument_reply	InstrumentsReply	Reply with a requested list of instruments.
instr_attrs_reply	InstrAttrsReply	Reply with instrument attributes in requested format.
search_reply	SearchInstrumentsReply	Reply to a search request with an instrument list.

5.1.39. SecsApiRequest

IRIS API request for information about securities.

The instrument_request, instr_attrs_request and search_request messages are mutually exclusive.

Sent to the queue: "jms.queue.iris.Secs" (see Section 3.20, “Securities and Instrument Attributes”).

Table 48. `SecsApiRequest` message fields

Name	Type	Description
serial_num	int64	Serial number of the request being sent.
instrument_request	InstrumentsRequest	Request for an instrument list.
instr_attrs_request	InstrAttrsRequest	Request to receive instrument attributes.
search_request	SearchInstrumentsRequest	Search request for an instrument list.

5.1.40. TotalsApiReply

IRIS API response with information about trades on securities market grouped for a period.

The totals_count_reply, totals_reply, quotations_reply, risers_fallers_reply, export_reply and error_message messages are mutually exclusive.

To receive this message, it is necessary to subscribe to the topic: "jms.topic.iris.Totals.client" (see Section 3.6, “Stock Market Trading Results”).

Table 49. `TotalsApiReply` message fields

Name	Type	Description
serial_num	int64	Serial number of the request to which this reply was received.
error_message	ErrorMessage	Reply with an error message.

totals_count_reply	TotalsCountReply	Reply with a count of trades records.
totals_reply	TotalsReply	Reply with grouped trading information.
quotations_reply	QuotationsReply	Reply with a list of quotations.
risers_fallers_reply	RisersFallersReply	Reply with information about rise and fall leaders.
Name	**Type**	**Description**
export_reply	TotalsExportReply	Reply with information about a formed export file with information about stock market trading results.

5.1.41. TotalsApiRequest

IRIS API request for information about trades on the securities market grouped for a period.

The count_request, totals_request, quotations_request, risers_fallers_request and export_request messages are mutually exclusive.

Sent to the queue: "jms.queue.iris.Totals" (see Section 3.6, “Stock Market Trading Results”).

Table 50. `TotalsApiRequest` message fields

Name	Type	Description
serial_num	int64	Serial number of the request being sent.
count_request	TotalsCountRequest	Request for a count of deals.
totals_request	TotalsRequest	Request for grouped trade information.
quotations_request	QuotationsRequest	Request for a list of quotations.
Risers_fallers_request	RisersFallersRequest	Request for information about rise and fall leaders.
export_request	TotalsExportRequest	Request for information about results of trades on the stock market as a file in the specified format.

5.1.42. WatchlistApiReply

IRIS API reply with general information about a list of user’s watchlist.

The search_reply, add_reply, del_reply, watchlist_reply and error_message messages are mutually exclusive.

To receive this message, it is necessary to subscribe to the topic:

"jms.topic.iris.Watchlist.client" (see Section 3.17, “User’s Watchlist”).

Table 51. `WatchlistApiReply` message fields

Name	Type	Description
serial_num	int64	Serial number of the request to which this reply was received.
error_message	ErrorMessage	Reply with an error message.

search_reply	WatchlistSearchReply	Reply with information about instruments upon addition to the user watchlist.
add_reply	WatchlistAddReply	Reply with information about results of addition of instruments to the user watchlist.
del_reply	WatchlistDelReply	Reply with information about results of removal of instruments from the user watchlist.
Name	**Type**	**Description**
watchlist_reply	WatchlistReply	Reply with information about quotations for insturmnets included into the user watchlist.

5.1.43. WatchlistApiRequest

IRIS API request to receive information about the user watchlist.

The search_request, add_request, del_request and watchlist_request messages are mutually exclusive.

Sent to the queue: "jms.queue.iris.Watchlist" (see Section 3.17, “User Watchlist”).

Table 52. `WatchlistApiRequest` message fields

Name	Type	Description
serial_num	int64	Serial number of the request being sent.
search_request	WatchlistSearchRequest	Rrequest to search for instruments upon addition to the user watchlist.
add_request	WatchlistAddRequest	Request to add instruments to the user watchlist.
del_request	WatchlistDelRequest	Request to remove instruments from the user watchlist.
watchlist_request	WatchlistRequest	Request to receive quotations for instruments included into the user watchlist.

5.2. Basic Types

5.2.1. CouponInfo

Extra information about coupon bond.

Table 53. `CouponInfo` message fields

Name	Type	Description
rate	Decimal	Coupon rate, % p.a.
min_rate	Decimal	Acceptable minimum rate, % p.a.

max_rate	Decimal	Acceptable maximum rate, % p.a.
maturity	Date	Repayment start date.
freq	int32	Frequency of coupon payment (times a yeara).
fixed_margin	Decimal	Fixed margin, % p.a.

5.2.2. Date

Date transfer structure.

Table 54. `Date` message fields

Name	Type	Description
day	int32	Day
month	int32	Month
year	int32	Year

5.2.3. DatePeriod

Date period transfer structure.

Table 55. `DatePeriod` message fields

Name	Type	Description
beg_date	Date	Period start date.
end_date	Date	Period end date.

5.2.4. DateTime

Date and time transfer structure.

Table 56. `DateTime` message fields

Name	Type	Description
date	Date	Day
time	Time	Time

5.2.5. Decimal

Decimal number transfer structure. Within IRIS API, it is supposedly a price or an indicator value.

Table 57. `Decimal` message fields

Name	Type	Description
value	string	Decimal value as a line.

5.2.6. DecimalRange

Transfer structure for a range of decimal numbers for using upon filtering within IRIS API.

Table 58. `DecimalRange` message fields

Name	Type	Description

beg	Decimal	Value from which the range startsн.
beg_inclusive	bool	Whether the value from which it starts is included in the range.
end	Decimal	Value with which the range ends.
end_inclusive	bool	Whether the value with which it ends is included in the range.

5.2.7. ErrorMessage

Message for transfer of information about an error that occurred on the server.

Table 59. `ErrorMessage` message fields

Name	Type	Description
code	ErrorMessageCode	Error message code.
message	string	Technical information about an error in text format.

5.2.8. GraphOhlc

Transfer structure for Open High Low Close (OHLC) – values of time, opening, maximum, minimum and closing of a period in format suitable for display on graphs.

Table 60. `GraphOhlc` message fields

Name	Type	Description
open	double	First parameter value for interva.
high	double	Highest parameter value for interva.
low	double	Lowest parameter value for interva.
close	double	Recent value of parameter for interval or current value of parameter where period is not yet closed.
volume	double	Volume for interval.
close_time	DateTime	Time at which the recent parameter value was fixed.

5.2.9. HTMLAttrsParams

Parameters of a request to receive object attributes where result must be obtained as a HTML-document.

Issuer or instrument can act as a request object.

Table 61. `HTMLAttrsParams` message fields

Name	Type	Description
Id	int32	Primary key of object's record into IRIS IS.

Lang	Language	Required language (as of 25.06.2024 for attributes of securities and instruments only EN and RU are supported, and for issuer attributes KZ, EN and RU are supported).

5.2.10. JSONAttrsParams

Parameters of a request to receive object attributes where result must be obtained as JSON.

Issuer or instrument can act as a request object.

Table 62. `JSONAttrsParams` message fields

Name	Type	Description
id	int32	Primary key of object's record into IRIS IS.
lang	Language	Required language (as of 25.06.2024 for attributes of securities and instruments only EN and RU are supported, and for issuer attributes KZ, EN and RU are supported).

5.2.11. Month

Month transfer structure.

Table 63. `Month` message fields

Name	Type	Description
month	int32	Number of month (starting from 1).
year	int32	Year.

5.2.12. MonthPeriod

Transfer structure for a period with grouping by months.

Table 64. `MonthPeriod` message fields

Name	Type	Description
beg_month	Month	Start of period.
end_month	Month	End of period.

5.2.13. Ohlc

Transfer structure for Open High Low Close (OHLC) – values of opening, maximum, minimum and closing for a required parameter for a definite period (see for instance Section 3.6, "Stock Market Trading Results").

Table 65. `Ohlc` message fields

Name	Type	Description
open	Decimal	First value of parameter for a period.
high	Decimal	Highest value of parameter for a period.

low	Decimal	Lowest value of parameter for a period.
close	Decimal	Last value of parameter for a period or current value of parameter, if the period is not year closed.
close_time	DateTime	Time during which the last parameter value was fixed.
wa	Decimal	Average weighted value of parameter for a period. In some cases, it may not be filled in (see for instance Section 5.12.3, “CurTotal”).
Name	**Type**	**Description**
trend	Decimal	Trend with respect to a next to the last value of the current period.
trend_ps	Decimal	Trend with respect to a next to the last value of the current period in percentage.
trend_time	DateTime	Time for trend values.

5.2.14. Orderbook

Order book entry.

Table 66. `Orderbook` message fields

Name	Type	Description
direction	OrderDirection	Direction or orders submitted at price set out in the “price” field.
price	Decimal	Price level of orders in order book. For repo operations, it is a yield level.
volume	int64	Order volume.

5.2.15. Quarter

Quarter transfer structure.

Table 67. `Quarter` message fields

Name	Type	Description
quarter	int32	Quarter number (1, 2, 3 or 4).
year	int32	Year.

5.2.16. QuarterPeriod

Transfer structure for a period with grouping by quarters.

Table 68. `QuarterPeriod` message fields

Name	Type	Description
beg_quarter	Quarter	Beginning of period.
end_quarter	Quarter	End of period.

5.2.17. Range

Transfer structure for a range in record list.

Table 69. `Range` message fields

Name	Type	Description
first	int32	Indext with which record range starts. First record has 0 index.
count	int32	Count of records in the range.

5.2.18. Time

Time transfer structure.

Table 70. `Time` message fields

Name	Type	Description
hour	int32	Hour
min	int32	Minute
sec	int32	Second

5.2.19. TotalsPeriod

Message to indicate required period for issue of grouped information about trades.

It is used to set a period when working with trades results and graphs (see Section 3.6, “Stock Market Trades Results”, Section 3.11, “Currency Market Trades Results”, Section 3.8, “Stock Market Graphs” and Section 3.13, “Currency Market Graphs”).

Table 71. `TotalsPeriod` message fields

Name	Type	Description
day	Date	Day. Mutually excluded with week, month, quarter, year, day_period, week_period, month_period, quarter_period, year_period.
week	Week	Week. Mutually excluded with day, month, quarter, year, day_period, week_period, month_period, quarter_period, year_period.
month	Month	Month. Mutually excluded with day, week, quarter, year, day_period, week_period, month_period, quarter_period, year_period.
quarter	Quarter	Quarter. Mutually excluded with day, week, month, year, day_period, week_period, month_period, quarter_period, year_period.
year	Year	Year. Mutually excluded with day, week, month, quarter, day_period, week_period, month_period, quarter_period, year_period.

day_period	DatePeriod	Period with grouping by days. Mutually excluded with day, week, month, quarter, year, week_period, month_period, quarter_period, year_period.
week_period	WeekPeriod	Period with grouping by weeks. Mutually excluded with day, week, month, quarter, year, day_period, month_period, quarter_period, year_period.
month_period	MonthPeriod	Period with grouping by months. Mutually excluded with day, week, month, quarter, year, day_period, week_period, quarter_period, year_period.
quarter_period	QuarterPeriod	Period with grouping by quarters. Mutually excluded with day, week, month, quarter, year, day_period, week_period, month_period, year_period.
year_period	YearPeriod	Period with grouping by years. Mutually excluded with day, week, month, quarter, year, day_period, week_period, month_period, quarter_period.

5.2.20. Week

Transfer structure for a number of week in year.

Table 72. `Week` message fields

Name	Type	Description
week	int32	Number of week in yeara.
year	int32	Year.

5.2.21. WeekPeriod

Transfer structure for a period of grouping by weeks.

Table 73. `WeekPeriod` message fields

Name	Type	Description
beg_week	Week	Beginning of period.
end_week	Week	End of period.

5.2.22. Year

Year transfer structure.

Table 74. `Year` message fields

Name	Type	Description
year	int32	Year.

5.2.23. YearPeriod

Transfer structure for a period of grouping by years.

Table 75. `YearPeriod` message fields

Name	Type	Description
beg_year	Year	Beginning of period.
end_year	Year	End of period.

5.2.24. ErrorMessageCode

Error message codes allowed when filling in the `code` field of the ErrorMessage message.

Table 76. `Allowed values of ErrorMessageCode`

Name	Description
EMC_UNKNOWN	An unexpected error occurred on the server. Application developers need to intervene.
EMC_BAD_USER	Error checking user rights: no username or password specified, invalid username or password.
EMC_BAD_REQUEST	Unknown type of incoming request.
EMC_BAD_PARAMS	Invalid parameters of the incoming request.
EMC_PROC_ERROR	Error processing the incoming request.

5.2.25. EventType

Type of operation with record (additing, changing, deleting).

Table 77. `Allowed values of EventType`

Name	Description
NEW	Adding a record.
UPDATE	Updating a record.
DELETE	Deleting a record.

5.2.26. ExportFileFormat

Export file format.

Table 78. `Allowed values of ExportFileFormat`

Name	Description
EFF_XLSX	Microsoft Excel.
EFF_XML	XML.
EFF_CSV	Comma-Separated Values (CSV).
EFF_JSON	JSON.

5.2.27. GraphIntervalType

Intervals supported upon issue of information for graphs.

Table 79. `Allowed values of GraphIntervalType`

Name	Description

GIT_5_MIN	5-minute interval.
GIT_15_MIN	15-minute interval.
GIT_30_MIN	30-minute interval.
GIT_HOUR	1-hour interval.
GIT_DAY	1-day interval.
GIT_WEEK	1-week interval.
GIT_MONTH	1-month interval.

5.2.28. Language

Languages supported in IRIS API.

Table 80. `Allowed values of Language`

Name	Description
RU	Russian lagnauge.
EN	English language.
KZ	Kazakh language.

5.2.29. LeaderType

Market leader type.

Table 81. `Allowed values of LeaderType`

Name	Description
LT_RISER	Rise leader.
LT_FALLER	Fall leader.

5.2.30. MarketSector

Market sector.

Table 82. `Allowed values of MarketSector`

Name	Description
PRIMARY	Primary (placement).
SECONDARY	Secondary.
SBS_SECTOR	Sale of state-owned shareholdings.
OFFERING	Special trades (placement). The sector is present for backward compatibility. It is not used because any placement is already a special trade, i.e. PRIMARY.
REPURCHASE	Special trades (repurchase).

5.2.31. OrderDirection

Direction of a submitted order (buy, sale).

Table 83. `Allowed values of OrderDirection`

Name	Description

BUY	Buy.
SELL	Sale.

5.2.32. OrderStatus

Status of submitted order.

Table 84. `Allowed values of OrderStatus`

Name	Description
ACTIVE	Active request.
PARTIALLY	Active but partially satisfied request.
FULLY	Fully satisfied.
CANCELED	Cancelled order before end of trade.

5.2.33. PriceType

Price type.

Table 85. `Allowed values of PriceType`

Name	Description
PT_MONEY	In money per unit.
PT_CLEAR	Clear price, in percentage of face value.
PT_DISCOUNT	Discount, in percentagebelow face value.
PT_SPREAD_BP	Spread (spread in basis points).
PT_SPREAD_PR	Price spread.
PT_YIELD	Yield.
PT_BP	Basis points.
PT_OTHER	Other.

5.3. General information

Determination of IRIS API messages intended for transfer of general information.

5.3.1. Answer

Answer in question.

Table 86. `Answer` message fields

Name	Type	Description
answer_id	int32	The field contains primary key of answer record in IRIS IS.
type	AnswerType	Answer type. It is filled in only on server. Upon answer generation, it may not be filled in.

text	string	Answer text. When obtaining an answer from the server, this field is always filled in. When generating an answer, the field is filled in only if answer type is QT_FREE.

5.3.2. Board

Reference data for MOEX regimes.

Table 87. `Board` message fields

Name	Type	Description
moex_boards_id	string	The field contains a primary key of MOEX regime in IRIS IS.
name	string	Regime name.

5.3.3. BoardsReply

Reply with a list of MOEX regimes.

Table 88. `BoardsReply` message fields

Name	Type	Description
boards	Board (array)	List of existing MOEX regimes.

5.3.4. BoardsRequest

Request for reference data under MOEX regimes.

Table 89. `BoardsRequest` message fields

Name	Type	Description
moex_boards_id	String	Primary key of MOEX regime. If no field is fixed, all existing MOEX regimes pop up.

5.3.5. Broker

Broker’s information.

Table 90. `Broker` message fields

Name	Type	Description
broker_id	int32	Broker’s code. The field contains primary key of broker’s record in IRIS IS.
name	string	Broker’s name.
logo	string	Reference to broker’s logo image which is stored on the server (URL).
contacts	string	Broker’s contact information.
site	string	Internet address of site.
email	string	Email address.

phone_numbers	string (array)	List of phone numbers.
offers	BrokerOffer (array)	List of advertisement offers from broker. List can be empty. List can contain not more than six sentences.

5.3.6. BrokerMessageReply

Reply to a message with request sent by user to broker.

Table 91. `BrokerMessageReply` message fields

Name	Type	Description
ok	bool	Status: `true` where request is successfully sent.
message	string	Field with reply message for user (optionally). Message will be sent by server in language set out in request in the `lang` field.

5.3.7. BrokerMessageRequest

Request to be sent by user to broker.

Table 92. `BrokerMessageRequest` message fields

Name	Type	Description
broker_id	int32	Code of the broker to whom the request will be sent. The field contains the primary key of the broker record in IRIS IS.
fio	string	Full name of user sending the request.
iin	string	IIN of user sending the request.
phone_number	string	Telephone number of user sending the request.
email	string	E-mail address of user sending the request.
Name	**Type**	**Description**
lang	Language	Language request.

5.3.8. BrokerOffer

Description of an advertisement offer from broker.

Table 93. `BrokerOffer` message fields

Name	Type	Description
image	String	Reference to an image which is stored on server (URL).

description	String	Description of a broker offer. Not more than 200 symbols.

5.3.9. BrokersListReply

Reply with a list of brokers.

Table 94. `BrokersListReply` message fields

Name	Type	Description
brokers	Broker (array)	List of brokers.

5.3.10. BrokersListRequest

Request for a list of brokers.

Table 95. `BrokersListRequest` message fields

Name	Type	Description
lang	Language	Request language.
broker_id	int32	Broker code. Optional field. To receive a full list of brokers, the field may not be filled in.

5.3.11. ChangePwdReply

Reply with password change result.

Table 96. `ChangePwdReply` message fields

Name	Type	Description
ok	bool	Status: `true` where password change was successful. Otherwise the **message** field will contain an error message.
message	string	Field with an error message.

5.3.12. ChangePwdRequest

Request for change of user password.

After request is successfully perfrormed, user can continue operation until session closes. Next connection must be created with a new password.

Table 97. `ChangePwdRequest` message fields

Name	Type	Description
new_password	String	New password.
lang	Language	Language of error messages.

5.3.13. MainListsReply

Message with information about current status of lists of instruments and indicators recommended for display to the end user.

Table 98. `MainListsReply` message fields

Name	Type	Description

main_shares_ids	int32 (array)	List of shares recommended for display to the end user first. The field contains primary keys of instrument records in IRIS IS.
main_cur_ids	int32 (array)	List of currency instruments recommended for display to the end user first. The field contains primary keys of currency instrument records in IRIS IS.
actual_cur_opers	CurOper (array)	List of current currency instruments that may be displayed to the end user. The field contains basic information about currency instruments in the form of CurOper list.
indicator_groups	IndicatorsGroup (array)	List of indicators recommended for display to the end user, divided into groups.
hide_empty_sections	bool	Parameter that determines whether empty sections should be hidden from the user (true - hide, false - not hide).
languages	Language (array)	List of languages supported in IRIS IS.
api_version	string	IRIS API version.

5.3.14. MainListsRequest

Request for information about current status of lists of instruments and indicators recommended for display to end user.

Table 99. `MainListsRequest` message fields

Name	Type	Description
lang	Language	Required reply language.

5.3.15. MarketRecord

Message for transfer of information about market in trades schedule.

Table 100. `MarketRecord` message fields

Name	Type	Description
market	ScheduleMarket	Market type in trades schedule.
name	string	Market name.
submarkets	SubmarketRecord (array)	Submarket table. Table must contain at least one record.

5.3.16. PoolAnswersReply

Reply to a message with sent poll answers.

Table 101. `PoolAnswersReply` message fields

Name	Type	Description
pool_id	int32	The field contains primary key of poll record in IRIS IS.
information	string	Information message from the server for display after successful completion of poll (optionally).

5.3.17. PoolAnswersRequest

Request for answers to the poll.

Table 102. `PoolAnswersRequest` message fields

Name	Type	Description
pool_id	int32	The field contains primary key of poll record in IRIS IS.
answers	Answer (array)	List of answers.

5.3.18. PoolInfo

Main information about conducted poll.

Table 103. `PoolInfo` message fields

Name	Type	Description
pool_id	int32	The field contains primary key of poll record in IRIS IS.
title	string	Short name of poll.

5.3.19. PoolQuestionsReply

Reply with questions of selected poll.

Table 104. `PoolQuestionsReply` message fields

Name	Type	Description
pool_id	int32	The field contains primary key of poll record in IRIS IS.
description	string	Detailed poll description.
Name	**Type**	**Description**
questions	Question (array)	List of questions.

5.3.20. PoolQuestionsRequest

Request for answers of the selected poll.

Table 105. `PoolQuestionsRequest` message fields

Name	Type	Description
pool_id	int32	The field contains primary key of poll record in IRIS IS.
lang	Language	Required poll language.

5.3.21. PoolsListReply

Reply with a list of active polls.

Table 106. `PoolsListReply` message fields

Name	Type	Description
pools	PoolInfo (array)	List of active requests sorted in required numerical order.

5.3.22. PoolsListRequest

Request for a list of active polls.

Table 107. `PoolsListRequest` message fields

Name	Type	Description
lang	Language	Required poll language.
section_id	int32	Primary key of column to which this poll is pegged (see Table 1, “News Messages Columns”). If no field is fixed, all valid polls pop up with no column limitation.

5.3.23. ProductInfo

IRIS information product.

Table 108. `ProductInfo` message fields

Name	Type	Description
product_id	int32	The field contains primary key of record of information product in IRIS IS (see Table 9, “Information Products”).
activated	Date	Start date of information provision.
deactivated	Date	End date of information provision. If the field is not filled in, there is no term limitation.

5.3.24. Question

Poll question.

Table 109. `Question` message fields

Name	Type	Description
question_id	int32	The field contains primary key of question record in IRIS IS.
type	QuestionType	Question type.
text	string	Question text.
answers	Answer (array)	Answer list.

5.3.25. SessionRecord

Information for transfer of information about session in the trades retulation table.

Table 110. `SessionRecord` message fields

Name	Type	Description
session_start	Time	Session start time.
session_end	Time	Session end time.
description	string	Session description

5.3.26. SubmarketRecord

Message for transfer of information about submarket in trades schedule.

Table 111. `SubmarketRecord` message fields

Name	Type	Description
name	string	**Name**
sessions	SessionRecord (array)	Table of trades schedule for submarket.

5.3.27. TradesScheduleReply

Reply with information about trades schedule.

Table 112. `TradesScheduleReply` message fields

Name	Type	Description
markets	MarketRecord (array)	List of markets for which information about trades schedule is obtained.

5.3.28. TradesScheduleRequest

Request of information about trades schedule.

Table 113. `TradesScheduleRequest` message fields

Name	Type	Description
lang	Language	Request language.
Name	**Type**	**Description**
market	ScheduleMarket	Market for which it is required to obtain trades status. If the field is not filled in, information will be returned for all markets.

5.3.29. TradesStatusReply

Reply with information about trades status.

Table 114. `TradesStatusReply` message fields

Name	Type	Description

trading_day	bool	Determining a status of current day by calendar: `true` – business day, `false` weekend or public holiday.
next_trading_day	Date	Determining a next trading day which follows the current.
trades_start	Time	Start time of trades for instrument.
trades_end	Time	End time of trades for instrument.

5.3.30. TradesStatusRequest

Request of information about trades status.

Table 115. `TradesStatusRequest` message fields

Name	**Type**	**Description**
lang	Language	Request language.
market	ScheduleMarket	Market for which it is required to obtain trades status.
instrument_id	int32	Financial in strument code.

5.3.31. UserFeedbackReply

Reply to user message sent to ensure feedback.

Table 116. `UserFeedbackReply` message fields

Name	**Type**	**Description**
ok	bool	Status: `true` if messafe is successfully sent.
message	string	Field with a replay message for user (optionally). Message will be sent by server in the language set out in request in the `lang` field.

5.3.32. UserFeedbackRequest

Message to be sent by user to ensure feedback.

Example of code to generate message in Java language:

```
InfoApiRequest request = InfoApiRequest.newBuilder()    .setUserFeedbackRequest(
      UserFeedbackRequest.newBuilder()
         .setFio("Surname Given name Patronymic name ")
         .setTel("+7 777 333 00 07")
         .setEmail("name@kase.kz")
         .setMessage("Problem description is set out here ")
         .setLang(Language.EN)
   ).build();
```

Table 117. `UserFeedbackRequest` message fields

Name	**Type**	**Description**
fio	string	User’s full name.

email	string	User’s e-mail.
tel	string	User’s phone number.
message	string	Message text.
lang	Language	Request language.

5.3.33. UserInfo

Information about IRIS uer.

Table 118. `UserInfo` message fields

Name	Type	Description
user_id	int32	The field contains primary key of user record in IRIS IS.
user_blocked	bool	Note of user blocking.
user_name	string	User’s name.
fio	string	User’s full name.
email	string	User’s e-mail.
client_id	int32	The field contains primary key of client record in IRIS IS.
client_blocked	bool	Note about client blocking.
client_name	string	Client’s name.
address	string	Client’s address.
tel	string	Client’s telephones.
demo_status	UserDemoStatus	Status of user registered for demo period.
demo_message	string	Message to user registered for demo period.

5.3.34. UserInfoReply

Reply with information about IRIS user.

Table 119. `UserInfoReply` message fields

Name	Type	Description
user_info	UserInfo	Information about IRIS user.
products	ProductInfo (array)	List of products available to the user. The field is to be filled in if `include_products_info` flag is specified in the request.

5.3.35. UserInfoRequest

Request of information about IRIS user.

Table 120. `UserInfoRequest` message fields

Name	Type	Description

include_products_info	bool	The parameter determines whether it is necessary to include in results information about information products available to the user.
lang	Language	Request language.

5.3.36. AnswerType

Answer type.

Table 121. `Allowed values of AnswerType`

Name	Description
AT_FIXED	Fixed answer type.
AT_FREE	Free answer type.

5.3.37. QuestionType

Question type.

Table 122. `Allowed values of QuestionType`

Name	Description
QT_ONE	Single answer type.
QT_MANY	Several answer options.
QT_FREE	One answer with free text.

5.3.38. ScheduleMarket

Market code in trades schedule.

Table 123. `Allowed values of ScheduleMarket`

Name	Description
SC_ALL	All categories.
SC_SHARES	Shares.
SC_BONDS	Corporate bonds.
SC_GC	Government securities.
SC_CURRENCY	Currency.

5.3.39. UserDemoStatus

Status of the user registered for demo period.

After registration, the user is assigned the `UDS_REGISTERED` status and can work with the terminal for 7 calendar days.

Starting from the third day after registration, user's status changes to `UDS_MESSAGE`. It is assumed that after each authorization, the client application using IRIS API will display a small unobtrusive window to the user with the notification text from the `demo_message` field.

On the eighth day after registration, the user is assigned the `UDS_BLOCKED` status. When a user with this status logs in, the client application must display a modal window that covers all content with the text from the `demo_message` field.

Table 124. `Allowed values of UserDemoStatus`

Name	Description
UDS_REGISTERED	The user has successfully registered for the demo period or does not belong to the demo user group.
UDS_MESSAGE	The user must receive a notification about the imminent end of the demo period.
UDS_BLOCKED	The demo period has ended. Access to information must be blocked.

5.4. Stock Market Deals

5.4.1. Deal

Information about securities market deals.

Table 125. `Deal` message fields

Name	Type	Description
deal_id	int64	Primary key of a deal record in IRIS IS.
deal_time	DateTime	Deal conclusion time.
instrument_id	int32	Primary key of instrument record in IRIS IS.
sec_code	string	Instrument code.
actual_sin	string	Valid identification number of security (NIN, ISIN, etc.)
buyer_id	int32	Primary key of counterparty buyer record in IRIS IS.
seller_id	int32	Primary key of counterparty seller in IRIS IS.
price	Decimal	Deal price.
price_unit	string	Price unit (currency code, per cents, face value units, etc.).
dirty_price	Decimal	Dirty deal price.
volume	int64	Deal volume.
vol_kzt	Decimal	Deal volume in KZT.
vol_usd	Decimal	Deal volume in USD at exchange rate accepted at KASE on a deal settlement day.
usd_rate	Decimal	Rate accepted at KASE on a deal settlement date.
Name	Type	Description
yield	Decimal	Yield.

state_id	int32	Primary key of deal status in IRIS IS (see Table 3, “Deal Statuses Codes”).
deal_type	DealType	Deal type.
deal_type_desc	string	Description of deal type (field is outdated and not filled in starting from version 2.10.0).
market_sector	MarketSector	Market sector.
market_sector_desc	string	Description of market sector (the field is outdated and is not filled in starting from version 2.10.0).
version	int64	Internal version of record in IRIS.
board	string	Trades regime code
currencies_id	int32	Primary key of instrument quotation currency in IRIS IS (see Table 5, “Currency Codes”).
sec_type	SecType	Security type.
coupon_info	CouponInfo	Additional information about coupon bond.
price_type	PriceType	Price type.

5.4.2. DealsBaseFilter

Information for filtering of a list of deals

Table 126. `DealsBaseFilter` message fields

Name	Type	Description
instruments_filter	InstrumentsFilter	Filter to fix limitations in the breakdown of instruments with which deals were made.
deal_type	DealType (array)	Listing of required types of deals.
market_sector	MarketSector (array)	Listing of required market sectors.
version_from	int64	Version starting from which records are required. This limitation is intended to be used in conjunction with the bandle size (see the "bandle" field in the `DealsRequest` message, Section 5.4.9, “DealsRequest”).
deals_ids	int64 (array)	List of deal codes for which information is required. It is intended to be used in conjunction with `DealsUpdate` (see Section 5.4.11, “DealsUpdate”).

confirmed	bool	If this field is filled in as `true`, only confirmed deals (code 1) will be returned. Otherwise, attention should be paid to the deal status transmitted in the "state_id" field of the `Deal` message (see Table 3, “Deals Status Codes”).

5.4.3. DealsCountReply

Reply with information about count of deals.

Table 127. `DealsCountReply` message fields

Name	Type	Description
Count	int32	Count of records qualified for filtering condition. Count to be returned accounts not only limitations fixed in filter but also limitations imposed on request performing user.

5.4.4. DealsCountRequest

Request for information about count of deals qualified for filtering conditions.

Note: Count to be returned accounts not only limitations fixed in filter but also limitations imposed on request performing user.

Table 128. `DealsCountRequest` message fields

Name	Type	Description
filter	DealsFilter	Filter for a requested list of deals.

5.4.5. DealsExportReply

Reply with information about formed export file.

Table 129. `DealsExportReply` message fields

Name	Type	Description
url	string	URL to download generated export file.

5.4.6. DealsExportRequest

Request for information about deals as a file in the specified format.

Table 130. `DealsExportRequest` message fields

Name	Type	Description
filter	DealsFilter	Filter for a requested list of deals.
sorting	DealsSortRecord (array)	Listing of fields for which sorting of requested list of deals is required.

format	ExportFileFormat	Required format of exported file.
lang	Language	Required language.

5.4.7. DealsFilter

Information to filter list of deals, including requested period of time.

Table 131. `DealsFilter` message fields

Name	Type	Description
period	DatePeriod	Period for which list of deals is requested.
deals_filter	DealsBaseFilter	Information to filter list of deals.

5.4.8. DealsReply

Reply with information about deals.

Table 132. `DealsReply` message fields

Name	Type	Description
range	Range	Range of recortds in reply.
deals	Deal (array)	List of deals

5.4.9. DealsRequest

Request for information about deals.

Table 133. `DealsRequest` message fields

Name	Type	Description
filter	DealsFilter	Filter for a requested list of deals.
range	Range	Indication of a required range of records. In reply to request, records that match the set filter and are included in the range specified in the parameter will be transmitted. Mutually exclusive with bandle.
bandle	int32	Indicaiton of a size of record packages to be transmitted. In reply to request, all records that match the set filter will be transmitted in packages of a size specified in the parameter. Mutually exclusive with range.
sorting	DealsSortRecord (array)	Listing of fields for which sorting of requisted list of deals is required.

5.4.10. DealsSortRecord

Information for sorting of a list of deals.

Table 134. `DealsSortRecord` message fields

Name	Type	Description
field	DealsSortField	Field for which sorting is required.
desc	bool	Sorting direction. If true is indicated, reverse sorting is required.

5.4.11. DealsUpdate

Update on occurrence of new deals or available changes.

Table 135. `DealsUpdate` message fields

Name	Type	Description
deals	ShortDealInfo (array)	Short information about deals.

5.4.12. ShortDealInfo

Short information about deal.

Table 136. `ShortDealInfo` message fields

Name	Type	Description
deal_id	int64	Deal code.
deal_time	DateTime	Deal time.
event_type	EventType	Event type (new, update, delete).
sec_type	SecType	Security type.

5.4.13. DealType

Deal type.

Table 137. `Allowed values of DealType`

Name	Description
TRADE	Market.
NEGO	Contractual.
OTC	Over the counter.

5.4.14. DealsSortField

Fields for which sorting of a list of deals is possible.

Table 138. `Allowed values of DealsSortField`

Name	Description
DSF_DEAL_TIME	Deal conclusion time.
DSF_INSTRUMENT	Instrument.
DSF_PRICE	Price.
DSF_DIRTY_PRICE	Dirty price deal.

DSF_VOLUME	Deal volume.
DSF_VOL_KZT	Deal volume in KZT.
DSF_YIELD	Yield.
DSF_VERSION	Record version.
DSF_ID	Record code in IRIS.
DSF_EXTID	External record code (identifier used upon import from BOS DB).

5.5. Stock Market Requests

Determination of IRIS API messages intended to transmit information about requests.

5.5.1. Order

Transfer structure for data about securities market order.

Table 139. `Order` message fields

Name	Type	Description
orders_id	int64	Primary key of order record in IRIS IS.
order_time	DateTime	Order submission time.
deltime	DateTime	Order delete time.
direction	OrderDirection	Direction of a submitted order (buy, sale).
status	OrderStatus	Order status.
type	OrderType	Order type.
instrument_id	int32	Primary key of instrument record in IRIS IS.
instrument_code	string	Instrument code.
actual_sin	string	Valid identificaiton number of security (NIN, ISIN, etc.)
currencies_id	int32	Primary key of currency of order submitted in IRIS IS (see Table 5, “Currency Codes”).
price	Decimal	Request price.
dirty_price	Decimal	Dirty request price.
yield	Decimal	Yield.
volume	int64	Request volume.
volrest	int64	Remaining request volume.
market_sector	MarketSector	Market sector.
market_sector_desc	string	Market sector description (the field is outdated and is not filled in starting

		from version 2.10.0).
version	int64	Internal version of record in IRIS.
board	string	Trades regime code.
vol_kzt	Decimal	KZT request volume.
vol_usd	Decimal	USD request volume at exchange rate accepted at KASE on the order submission date.
sec_type	SecType	Security type.
coupon_info	CouponInfo	Additional information about coupon bond.
price_type	PriceType	Price type.

5.5.2. OrdersCountReply

Reply with information deals count.

Table 140. `OrdersCountReply` message fields

Name	Type	Description
count	int32	Count of records that match the filtering condition. Count to be returned accounts not only limitations fixed in filter but also limitations imposed on request performing user.

5.5.3. OrdersCountRequest

Request for information about count of orders that match the filtering conditions.

Note: Count to be returned accounts not only limitations fixed in filter but also limitations imposed on request performing user.

Table 141. `OrdersCountRequest` message fields

Name	Type	Description
filter	OrdersFilter	Filter for a requested order list.

5.5.4. OrdersExportReply

Reply with information about formed export file.

Table 142. `OrdersExportReply` message fields

Name	Type	Description
url	string	URL to download generated export file.

5.5.5. OrdersExportRequest

Request for information about requests as a file in the specified format.

Table 143. `OrdersExportRequest` message fields

Name	Type	Description
filter	OrdersFilter	Filter for a requested order list.
sorting	OrdersSortRecord (array)	Listing of fields for which sorting of a requested order list is required.
format	ExportFileFormat	Required format of export file.
lang	Language	Required language.

5.5.6. OrdersFilter

Information about filtering of order list.

Table 144. `OrdersFilter` message fields

Name	Type	Description
period	DatePeriod	Period for which order list is requested.
Name	**Type**	**Description**
instruments_filter	InstrumentsFilter	Filter to fix limitations in the breakdown of instrumentsw with which orders were made.
direction	OrderDirection (array)	Citation of required order directions. If no required directions are set, orders of all directions are issued.
order_type	OrderType (array)	Citation of required order types. If no required types are set, all order types are issued.
market_sector	MarketSector (array)	Citation of required market serctors. If no required sectors are set, orders from all sectors are issued.
version_from	int64	Version starting from which records are required. It is assumed that this limitation will be used in combination with specification of package size (see the “bandle” field in the `OrdersRequest` message, Section 5.5.8, “OrdersRequest”).
orders_ids	int64 (array)	List of order codes for which it is required to obtain information. It is assumed to be used in combination with `OrdersUpdate` (see Section 5.5.10, “OrdersUpdate”).

5.5.7. OrdersReply

Reply with information about orders.

Table 145. `OrdersReply` message fields

Name	Type	Description
range	Range	Range of orders in reply.
orders	Order (array)	Order list.

5.5.8. OrdersRequest

Request for information about requests.

Table 146. `OrdersRequest` message fields

Name	Type	Description
filter	OrdersFilter	Filter for a requested order list.
range	Range	Indication of a required record range. In reply to request, records that match the set filter and are included into the range specified in the parameter will be transmitted. Mutually exclusive with bandle.
bandle	int32	Indication of a size of record package to be transmitted. In reply to request, All records that match the set filter and are included into the range specified in the parameter will be transmitted. Mutually exclusive with range.
sorting	OrdersSortRecord (array)	Listing of fields for which sorting of a requested order list is required.

5.5.9. OrdersSortRecord

Information to sort order list.

Table 147. `OrdersSortRecord` message fields

Name	Type	Description
field	OrdersSortField	Field under which sorting is required.
desc	bool	Sorting direction. If true is specified, reverse sorting is required.

5.5.10. OrdersUpdate

Notice about occurrence of new orders or available changes.

Table 148. `OrdersUpdate` message fields

Name	Type	Description
orders	ShortOrderInfo (array)	Short information about orders.

5.5.11. ShortOrderInfo

Short information about order

Table 149. `ShortOrderInfo` message fields

Name	Type	Description
order_id	int64	Order code.
order_time	DateTime	Order time.
event_type	EventType	Event type (new, update, delete).
sec_type	SecType	Security type.

5.5.12. OrderType

Type of submitted order.

Table 150. `Allowed values of OrderType`

Name	Description
LIMIT	Request is limited for open trades method.
MARKET	Market order.
NEGO_ORDER	Order is limited for direct deals method.

5.5.13. OrdersSortField

Fields for which order list is possible.

Table 151. `Allowed values of OrdersSortField`

Name	Description
OSF_ORDER_TIME	Order submission time.
OSF_INSTRUMENT	Instrument.
OSF_PRICE	Price.
OSF_DIRTY_PRICE	Dirty order price.
OSF_VOLUME	Order volume.
OSF_YIELD	Yield.
OSF_VERSION	Record version.
OSF_ID	Record code in IRIS.
OSF_EXTID	External record code (identifier used upon import from BOS DB).

5.6. Issuer Attributes

5.6.1. Issuer

Basic issuer information.

Table 152. `Issuer` message fields

Name	Type	Description
issuer_id	int32	Primary key of issuer’s record in IRIS IS.
code	string	Issuer’s code.
status	IssuerRecordStatus	Status of record with issuer’s information.
name	string	Issuer’s name.
instrument_ids	int32 (array)	List of primary keys of issuer’s securities. It is filled in only where it was clearly requested (see the `include_sec_ids` field in IssuersRequest).
version	int64	Internal version of record in IRIS IS.

5.6.2. IssuerAttrsReply

Reply with issuer attributes.

Table 153. `IssuerAttrsReply` message fields

Name	Type	Description
issuer_id	int32	Primary key of issuer’s record in IRIS IS.
html	string	Issuer’s attributes in HTML format. It is filled in where HTMLAttrsParams type was transmitted in parameters of IssuerAttrsRequest message.
json	string	Issuer’s attributes in JSON format. It is filled in where JSONAttrsParams type was tramsferred in parameters of IssuerAttrsRequest message. Format of JSON-file to be returned is described in section of reference tables (see. Section 4.1, “Structure of JSON-fille with issuer’s attributes”).

5.6.3. IssuerAttrsRequest

Request to receive issuer’s attributes.

Table 154. `IssuerAttrsRequest` message fields

Name	Type	Description
html	HTMLAttrsParams	Request parameter where result must be obtained in HTML format. Mutually exclusive with parameter json.

json	JSONAttrsParams	Request parameter where result must be obtained in JSON format. Mutually exclusive with parameter html.

5.6.4. IssuersFilter

Information to filter issuer list.

Table 155. `IssuersFilter` message fields

Name	Type	Description
codes	string (array)	List of issuer codes required for download. Both codes and templates (see Section 3.22, “Filtering Templates”) can be transmitted for filtering.
names	string (array)	List of issuer names required for download. Both codes and templates (see Section 3.22, “Filtering Templates”) can be transmitted for filtering.
isin	string	Field for issuer search by ISIN of one of securities it owns (for older securities, NIN or another possible identifier can be used in search).
ids	int32 (array)	List of primary issuer keys required for loading.
status	IssuerRecordStatus (array)	List of required record statuses. Where the list is empty, all records without limitation will be issued.
Name	**Type**	**Description**
version	int64	Internal version of record in IRIS IS starting from which result must be issued.

5.6.5. IssuersReply

Reply with a list of requested issuers.

Table 156. `IssuersReply` message fields

Name	Type	Description
range	Range	Range of reply records.
issuers	Issuer (array)	Issuer list.

5.6.6. IssuersRequest

Request for a list of issuers.

Table 157. `IssuersRequest` message fields

Name	Type	Description
filter	IssuersFilter	Filter for requested issuer list.
range	Range	Indication of required record range. Records that match the set filter and are included into the range specified in parameter will be transmitted as a reply to the request. Mutually exclusive with bandle.
bandle	int32	Indication of size of record packages to be transmitted. All records that match the set filter will be transmitted in packages of a size set out in the filter. Mutually exclusive with range.
include_sec_ids	bool	Where this field is filled in as `true`, the `instrument_ids` list will be filled in the Issuer messages that arrived in reply.
lang	Language	Required reply language (as of 18.07.2022 only EN and RU are supported).

5.6.7. IssuerRecordStatus

Status of record with issuer information.

Table 158. `Allowed values of IssuerRecordStatus`

Name	Type
IRS_ACTIVE	Active.
IRS_HISTORY	Historic.

5.7. Securities Attributes

5.7.1. InstrAttrsReply

Reply with instrument attributes.

Table 159. `InstrAttrsReply` message fields

Name	Type	Description
instrument_id	int32	Primary key of instrumenet record in IRIS IS.
html	string	Instrument attributes as HTML-document. Document contains a set of attributes valid at the time of order execution.

5.7.2. InstrAttrsRequest

Request to receive instrument attributes.

Table 160. `InstrAttrsRequest` message fields

Name	Type	Description
html	HTMLAttrsParams	Request parameter where result must be obntained as HTML-document. HTML-document to b e returned will contain instrument attributes in combination with security attributes.

5.7.3. Instrument

Basic instrument information.

Table 161. `Instrument` message fields

Name	Type	Description
instrument_id	int32	Primaray key of instrument record in IRIS IS.
security_id	int32	Primary key of security record in IRIS IS.
code	string	Instrument code.
status	SecurityRecordStatus	Record status with security information.
type	SecType	Security type.
trade_area	TradeArea	Trade area.
government	bool	Whether the traded security is a government security.
issuer_info	Issuer	Issuer information. It is filled in only where it is expressly requested see the `include_issuer_info` field in InstrumentsRequest).
version	int64	Internal version of record in IRIS IS.

5.7.4. InstrumentsFilter

Information to filter list of securities and instruments.

Conditions formed based on `codes`, `isin` and `ids` fields are combined by “OR”. Other conditions are combined by “AND”.

Table 162. `InstrumentsFilter` message fields

Name	Type	Description

codes	string (array)	List of codes of securities and instruments required for download. Both codes and templates can be transmitted for filtering (see Section 3.22, "Filtering Templates"). Length of the code transmitted in the request must not be less than 4 characters.
isin	string	Field for searching for an instrument by its ISIN identifier (as well as by NIN for older securities).
ids	int32 (array)	List of primary keys required for loading.
status	SecurityRecordStatus (array)	List of required record statuses. If the list is empty, all records will be returned without limitation.
sec_type	SecType (array)	List of required types of securities. When forming a request, types of securities are combined by "OR". When adding a condition for the BONDS type, the government field is also taken into account.
government	GovernmentSelector	Attribute to filter government securities. This field is only relevant for the `BONDS` security type (see Section 5.7.11, "SecType")
trade_area	TradeArea	Required trade area.
issuer_id	int32	Search by primary issuer key.
version	int64	Internal version of record in IRIS IS starting from which result must be issued.
index_kase	bool	If this parameter is set in `true`, list of securities will be limited by securities included in a representative list of KASE Index.

5.7.5. InstrumentsReply

Reply with a list of requested instruments.

Table 163. `InstrumentsReply` message fields

Name	Type	Description
range	Range	Record range in reply.
instruments	Instrument (array)	List of instruments sorted by instrument code.

5.7.6. InstrumentsRequest

Request for an instrument list.

Table 164. `InstrumentsRequest` message fields

Name	Type	Description
filter	InstrumentsFilter	Filter for requested instrument list.
range	Range	Indication of a required range of records. Reply to the request will include records that match the set filter and are included in the range specified in the parameter. Mutually exclusive with bandle.
bandle	int32	Indication of a size of record packages to be transmitted. In reply to the request, all records that match the set filter will be transmitted in packages of a size specified in the parameter. Mutually exclusive with range.
include_issuer_info	bool	Where this field is filled in as `true` in the Instrument messages that came in reply, the `issuer_info` field will be filled in.
lang	Language	Required language (as of 18.07.2022 only EN and RU are supported). The field should be filled only where issuer information was requested (include_issuer_info=true).

5.7.7. SearchInstrumentsReply

Reply to search request with an instrument list (see Section 5.7.8, “SearchInstrumentsRequest”).

Table 165. `SearchInstrumentsReply` message fields

Name	Type	Description
range	Range	Record range in reply.
instruments	Instrument (array)	List of instruments sorted by instrument code.

5.7.8. SearchInstrumentsRequest

Search request for an instrument list. Unlike InstrumentsRequest, in this request InstrumentsFilter filter is replaced by line on which search by codes of financial instruments and also by codes and names of issuers is made.

Table 166. `SearchInstrumentsRequest` message fields

Name	Type	Description
search_query	string	Line that contains a search query.

Name	Type	Description
shares	bool	Whether to include shares in results.
bonds	bool	Whether to include corporate bonds in results.
government	bool	Whether to include government securities in results.
include_issuer_info	bool	Where this field is filled in as `true` in the Instrument messages that came in reply, the `issuer_info` field will be filled in.
range	Range	Indication of a required record range. In reply to request records that match the set filter and are included in the range specified in the parameter will be transmitted. Mutually exclusive with bandle.
bandle	int32	Indication of a size of record bandles to be transmitted. In reply to request all records that match the set filter will be transmitted in packagegs of a set specified in the parameter. Mutually exclusive with range.
lang	Language	Required language (as of 18.07.2022 only EN and RU are supported). The field should be filled only where issuer information was requested (include_issuer_info=true).

5.7.9. Security

Basic security information.

Table 167. `Security` message fields

Name	Type	Description
security_id	int32	Primary key of security record in IRIS IS.
code	string	Security code.
status	SecurityRecordStatus	Record status with security information.
type	SecType	Security type.
government	bool	Whether the security is a government one.
instruments	int32 (array)	Primary keys of insturment records in IRIS IS.

version	int64	Internal version of record in IRIS IS.

5.7.10. GovernmentSelector

Listed type to filter government securities.

Table 168. `Allowed values of GovernmentSelector`

Name	Description
GS_ALL	Result must return all securities, irrespective of their belonging to government ones.
GS_GOVERNMENT	Result must return only government securities.
GS_CORPORATE	Result must return only corporate securities.

5.7.11. SecType

Security type.

Table 169. `Allowed values of SecType`

Name	Description
SHARES	Shares.
BONDS	Bonds.
SBS	SBS.
UNIT	Units.
FUTURES	Futures.
ETF	ETF.
GDR	Depository receipts.
CPC	Clearing participation certificates (CPC).

5.7.12. SecurityRecordStatus

Record status with security information.

Table 170. `Allowed values of SecurityRecordStatus`

Name	Description
SRS_ACTIVE	Active. This status implies that security is in listing.
SRS_HISTORY	Historic.
SRS_DELETED	Deleted. Records with this studies should be ignored when working with IRIS IS archives.

5.7.13. TradeArea

Trade area.

Table 171. `Allowed values of TradeArea`

Name	Description
KASE	KASE (valid).
RFCA	RFCA (historic data).

5.8. Results of tradeds in Securities

5.8.1. QuotationsReply

Reply with information about quotations.
For quotations, in the Total message, the following fields are fileld in: instrument_id, instrument_code, sec_type, price.open, price.high, price.low, price.close, price.close_time, price.wa, price.trend, price.trend_ps, volume.close, kzt_sum, usd_sum, deal_count.

Table 172. `QuotationsReply` message fields

Name	Type	Description
totals	Total (array)	Quotation list.

5.8.2. QuotationsRequest

Request for information about quotations. Each record to be returned by request will contain information about the last trade in the instrument, the trend, and a limited set of grouped data for the current day compared to TotalsReply.

Regardless of the specified filter, quotation data is always based on confirmed market trades on the secondary market, which is similar to confirmed=true, deal_type=TRADE and market_sector=SECONDARY in DealsBaseFilter.

Table 173. `QuotationsRequest` message fields

Name	Type	Description
filter	DealsBaseFilter	Limitation on a list of requested instruments.

5.8.3. RisersFallersReply

Reply with information about rise and fall leaders.

In the Total field, for rise and fall leaders the following fields are filled in: instrument_id, instrument_code, price.close, price.close_time, price.trend_ps.

Table 174. `RisersFallersReply` message fields

Name	Type	Description
leader_type	LeaderType	Market leader type in the result.
sec_type	SecType	Securities type in the result.
government	GovernmentSelector	Attributes to determine government securities.
totals	Total (array)	List of rise and fall leaders (depending on leader_type).

5.8.4. RisersFallersRequest

Request for information about rise and fall leaders.

Data about rise and fall leaders are based on confirmed market deals on the secondary market, which is similar to confirmed=true, deal_type=TRADE and market_sector=SECONDARY in DealsBaseFilter.

Table 175. `RisersFallersRequest` message fields

Name	Type	Description

leader_type	LeaderType	Required market leader type.
depth	int32	Depth of rise and fall leader list.
sec_type	SecType	Required securities type. When `BONDS` type is indicated, the `government` field is also taken into account.
Name	**Type**	**Description**
government	GovernmentSelector	Attribute for filtering of government securities. This field has a value only for `BONDS` security type (see Section 5.7.11, “SecType”)

5.8.5. Total

Information about trades on the stock market grouped for a period by instrument, market secrtor and deal type.

Table 176. `Total` message fields

Name	**Type**	**Description**
instrument_id	int32	Primary key of instrument record in IRIS IS.
market_sector	MarketSector	Market sector.
deal_type	DealType	Deal type.
instrument_code	string	Instrument code.
sec_type	SecType	Security type.
Price	Ohlc	OHLC-information about price.
Volume	Ohlc	OHLC-information about volume of KZT deals.
kzt_sum	Decimal	Aggregate volume of trades for a period in KZT.
kzt_sum_trend	Decimal	Trend for an aggregate trading volume for a period in KZT compared to a previous period.
kzt_sum_trend_ps	Decimal	Trend for an aggregate trading volume for a period in KZT compared to a previous period in percentage.
usd_sum	Decimal	Aggregate trading volume for a period in USD.
prev_period	TotalsPeriod	Previous period for aggregage trading volumes.
deal_count	int32	Cound of deals for a period.
order_count	int32	Cound of orders for a period.
government	bool	Whether the security is a government one.

period	TotalsPeriod	Period for which information is grouped (starting from version 2.10.07).
volume_sum	int64	Aggregate volume of trades for a period in instrument count (from version 2.11.09).
actual_sin	string	Valid security identificaiton number (NIN, ISIN, etc., starting from version 2.11.21).

5.8.6. TotalsCountReply

Reply with information about deal count.

Table 177. `TotalsCountReply` message fields

Name	Type	Description
count	int32	Count of records that match the condition set in the request.

5.8.7. TotalsCountRequest

Request for information about count of deals that match the filtering condition.

Note: Count to be returned accounts not only limitations fixed in filter but also limitations imposed on request performing user.

Note: Count request cannot be performed for periods which represent a range (i.e. day_period, week_period, month_period, quarter_period and year_period).

Table 178. `TotalsCountRequest` message fields

Name	Type	Description
filter	TotalsFilter	Filter for a requested list in the breakdown of deals of interest. Limitations for orders to calculate bid and offer will be determined subject to deal limitations.

5.8.8. TotalsExportReply

Reply with information about generated export file with information about stock market trades.

Table 179. `TotalsExportReply` message fields

Name	Type	Description
url	string	URL to download generated export file.

5.8.9. TotalsExportRequest

Request for information about stock market trades as a file in the specified format.

Table 180. `TotalsExportRequest` message fields

Name	Type	Description

filter	TotalsFilter	Filter for a requested list in the breakdown of deals of interest. Limitations for orders to calculate bid and offer will be determined subject to deal limitations.
format	ExportFileFormat	Required format of export file.
lang	Language	Required language.

5.8.10. TotalsFilter

Information to filter trades data.

When filling in DealsBaseFilter, please note that the version_from, deals_ids and confirmed fields are not used in this context. Daily totals are generated based on all confirmed deals.

Likewise, the version field in InstrumentsFilter is not used.

Table 181. `TotalsFilter` message fields

Name	Type	Description
period	TotalsPeriod	Required period for data grouping.
filter	DealsBaseFilter	Filter for a requested list in terms of deals of interest. Limits on orders to calculate bid and offer will be determined subject to limitations on deals. Trading results do not include data on over-the-counter deals. As a result, including the `OTC` deal type in the filter conditions (see Section 5.4.13, “DealType”) does not make sense with respect to stock market trading results. Due to specifics of requests for totals, the version_from, deals_ids and confirmed fields are also not used in DealsBaseFilter. In InstrumentsFilter, fields status, trade_area, issuer_id, version and index_kase are not used for the same reason.

all_instruments	bool	The field is not used from version 2.11.41. The QuotationsRequest request has been added to obtain similar information. If the field is filled as `false` (default value), totals will contain only those instruments that match the filter and were traded in the requested period. If the field is filled as `true`, the request will return all instruments that match the filter. For those instruments that were not traded in the requested period, the last deal will be found, data on which will be filled in the `price.close`, `price.close_time`, `volume.close` fields (in addition to the i`nstrument_id`, `instrument_code`, `market_sector`, `deal_type`, `sec_type` fields).

5.8.11. TotalsReply

Reply with information about trades grouped for a period.

Table 182. `TotalsReply` message fields

Name	Type	Description
period	TotalsPeriod	Period for which information is grouped.
range	Range	Record range in reply.
totals	Total (array)	List of records with grouped trading information.

5.8.12. TotalsRequest

Request for information about stock market trades.

Table 183. `TotalsRequest` message fields

Name	Type	Description
filter	TotalsFilter	Filter for a requested list in terms of deals of interest. Limitations for orders to calculate bid and offer will be determined subject to deals limitations.
range	Range	Indication of a required record range. In reply to request records that match the set filter and are included in the range specified in the parameter will be transmitted. Mutually exclusive with bandle.

bandle	int32	Indication of a size of record bandles to be transmitted. In reply to request all records that match the set filter will be transmitted in packagegs of a set specified in the parameter. Mutually exclusive with range.

5.8.13. TotalsUpdate

Update on occurrence of grouped information about trandes on the stock market.

Table 184. `TotalsUpdate` message fields

Name	Type	Description
periods	TotalsPeriod (array)	List of periods for which trades information is changed.

5.9. News

5.9.1. File

Message to transmit file attributes.

Table 185. `File` message fields

Name	Type	Description
files_id	int32	Primary file key.
Name	**Type**	**Description**
created	DateTime	File creation date and time.
filetypes_id	int32	File type (see Table 2, “File types”).
filename	string	File name.
crc32	int64	Control file sum (CRC-32).
size	int32	File size in bytes.
description	string	File description. Optionally.
url	string	URL for file download. Optionally.

5.9.2. FileBody

File or its part as an array of bytes in combination with attributes.

Table 186. `FileBody` message fields

Name	Type	Description
file	File	Attributes of requested file.
body	bytes	Whole file or its part as a byte array.

5.9.3. FileBodyReply

Reply with file or its part as a byte array.

Table 187. `FileBodyReply` message fields

Name	Type	Description
file	File	Attributes of requested file.
body	bytes	Whole file or its part as a byte array.
range	Range	It determines what part of the file is transmitted to the “body”. If no “range” is set, the whole file is transmitted to “body”.

5.9.4. FileBodyRequest

Request to receive file by its primary key.

It is supposed to be used with primary key file which was received in `NewsRefsReply` in reply to the `NewsRefsRequest` sending (see Section 5.9.18, “NewsRefsReply” and Section 5.9.17, “NewsRefs”). The “range” and “bandle” fields in “partial_load” determine how the file will be transmitted. If none of these fields is set, the file will be transmitted in whole in one message.

Table 188. `FileBodyRequest` message fields

Name	Type	Description
files_id	int32	Primary key of requested file.
range	Range	Indication of a required range of bytes. Count of bytes specified in the “count” will be transmitted, starting from a byte specified in “star” (see Section 5.2.17, “Range”).
Name	**Type**	**Description**
		Mutually exclusive with bandle.
bandle	int32	Indication of a size of blocks to be transmitted. File will be transmitted in several `FileBodyReply` messages, where each will contain as many bytes as it is specified in “bandle” (see Section 5.9.3, “FileBodyReply”). Mutually exclusive with range.

5.9.5. FilesReply

Reply with a list of files. The message transfers only file attributes. To receive a file in binary form, it is necessary to seld `FileBodyRequest` message (see Section 5.9.4, “FileBodyRequest”).

Table 189. `FilesReply` message fields

Name	Type	Description
files	File (array)	File list.

5.9.6. FilesRequest

Request to receive file attributes linked to a news message by their primary key of news message.

It is supposed to use the `files_ids` attribute with primary keys which were received in `NewsRefsReply` in response to the `NewsRefsRequest` sending (see Section 5.9.18, “NewsRefsReply” and Section 5.9.17, “NewsRefs”).

Table 190. `FilesRequest` message fields

Name	Type	Description
news_id	int32	Primary record key of a news message in IRIS IS.
files_ids	int32 (array)	Primary keys of required files.

5.9.7. FullNewsInfo

Full information about a news message. It is used to reduce count of requests to server.

Table 191. `FullNewsInfo` message fields

Name	Type	Description
news	News	News message attributes.
body	string	News message body in HTML format.
files	FileBody (array)	Attached files (body and attributes).

5.9.8. FullNewsInfoReply

Reply with full information about news messages.

Table 192. `FullNewsInfoReply` message fields

Name	Type	Description
news	FullNewsInfo (array)	List of news messages.

5.9.9. FullNewsInfoRequest

Request to receive full information about news messages. It is used to reduce count of requests to server.

Table 193. `FullNewsInfoRequest` message fields

Name	Type	Description
filter	NewsFilter	Filter for a requested list of news messages.
sorting	NewsSortRecord (array)	Listing of fields for which sorting of a requested list of deals is required.

5.9.10. News

News message.

Table 194. `News` message fields

Name	Type	Description
news_id	int32	Primary record key of a news message in IRIS IS.

news_time	DateTime	News message release date.
title	string	News message heading.
lang	Language	News message language.
beg_time	DateTime	Event beginning date and time.
end_time	DateTime	Event end date and time.
eventplace	string	Event place description.
source_id	int32	Source code of a news message.
comments	string	Comments to a news message.
version	int64	Internal version of record in IRIS IS.
selected	bool	If true is specified, news is marked as selected.

5.9.11. NewsBody

News message body.

Table 195. `NewsBody` message fields

Name	Type	Description
news_id	int32	Primary record key of a news message in IRIS IS.
Body	string	News message body in HTML format.

5.9.12. NewsBodyReply

Reply with a list of news messages.

Table 196. `NewsBodyReply` message fields

Name	Type	Description
bodies	NewsBody (array)	List of news messages.

5.9.13. NewsBodyRequest

Request to receive selected news m essages.

It is supposed to be used with codes of news messages which were received in `NewsReply` in reply to the `NewsRequest` sending (see Section 5.9.21, “NewsRequest” and Section 5.9.20, “NewsReply”) .

Table 197. `NewsBodyRequest` message fields

Name	Type	Description
news_ids	int32 (array)	List of codes of news messages for which it is required to obtain information.

5.9.14. NewsCountReply

Reply with information about news message count.

Table 198. `NewsCountReply` message fields

Name	Type	Description
count	int32	Count of records that match the filtering condition. Note: Count to be returned accounts not only limitations fixed in filter but also limitations imposed on request performing user.

5.9.15. NewsCountRequest

Request for information about count of news messages that match the filtering conditions.

Note: Count to be returned accounts not only limitations fixed in filter but also limitations imposed on request performing user.

Table 199. `NewsCountRequest` message fields

Name	Type	Description
filter	NewsFilter	Filter for a requested list of news messages.

5.9.16. NewsFilter

Information to filter a list of news messages.

Table 200. `NewsFilter` message fields

Name	Type	Description
period	DatePeriod	Period for which a list of news messages is requested.
newssections	int32 (array)	List of codes of columns to which a news message must be linked. Codes set out in this parameter are combined by `OR`.
secs	int32 (array)	List of codes of securities to which a news message must be linked.
issuers	int32 (array)	List of codes of issuers to which a news message must be linked.
lang	Language (array)	Required languages of a news message.
fts_condition	string	Condition for a full-text search. Request must be formed subject to rules described in section “`Boolean Full-Text Searches`”[6] of documentation under MySQL.

[6] https://dev.mysql.com/doc/refman/5.6/en/fulltext-boolean.html

version_from	int64	Version starting from which records are required. It is supposed that this limitation will be used in combimation with indication of a bandle size (see the “bandle” field in the `NewsRequest` message, Section 5.9.21, “NewsRequest”).
news_ids	int32 (array)	List of codes of news messages for which it is required to obtain information. It is supposed to use in combination with `NewsUpdate` (see Section 5.9.23, “NewsUpdate”).
only_selected	bool	Selected news. If true is indicated, only selected news will be transmitted.

5.9.17. NewsRefs

News message links. News message can be linked to issuers, securities and rubricator and also have attached files).

Table 201. `NewsRefs` message fields

Name	Type	Description
news_id	int32	Primary record key of a news message in IRIS IS.
newssections_ids	int32 (array)	List of primary keys of columns to which a news message is linked (see Table 1, “Columns of news messages”).
secs_ids	int32 (array)	List of primary keys of securities to which a news message is linked.
issuers_ids	int32 (array)	List of primary keys of issuers to which a news message is linked.
files_ids	int32 (array)	List of primary keys of files attached to a news message.

5.9.18. NewsRefsReply

Reply with links for selected news messages.

Table 202. `NewsRefsReply` message fields

Name	Type	Description
refs	NewsRefs (array)	List with links for selected news messages.

5.9.19. NewsRefsRequest

Request to receive links for selected news messages.

It is supposed to use bews messages which were received in `NewsReply` in response to the `NewsRequest` sesnding (see Section 5.9.21, “NewsRequest” and Section 5.9.20, “NewsReply”) .

Table 203. `NewsRefsRequest` message fields

Name	Type	Description
news_ids	int32 (array)	List of codes of news messages for which it is required to receive information.

5.9.20. NewsReply

Reply with a list of titles of news messages.

Table 204. `NewsReply` message fields

Name	Type	Description
range	Range	Range of records in reply.
news	News (array)	List of news messages.

5.9.21. NewsRequest

Request for a list of titles of news messages.

Table 205. `NewsRequest` message fields

Name	Type	Description
Filter	NewsFilter	Filter for a requested list of news messages.
Range	Range	Indication of a required record range. In reply to request records that match the set filter and are included in the range specified in the parameter will be transmitted. Mutually exclusive with “bandle”.
Name	**Type**	**Description**
bandle	int32	Indication of a size of record bandles to be transmitted. In reply to request all records that match the set filter will be transmitted in packagegs of a set specified in the parameter. Mutually exclusive with “range”.
sorting	NewsSortRecord (array)	Listing of fields under which sorting of a requested list of deals is required.

5.9.22. NewsSortRecord

Information to sort a list of news messages.

Table 206. `NewsSortRecord` message fields

Name	Type	Description
field	NewsSortField	Field under which sorting is required.

desc	bool	Sorting direction. If true is indicated, reverse sorting is required.

5.9.23. NewsUpdate

Update on occurrence of new news messages or available changes.

Table 207. `NewsUpdate` message fields

Name	Type	Description
news	ShortNewsInfo (array)	Short information about a news message.

5.9.24. NewsUsersAddReply

Reply to a request to add news to a selected list

Table 208. `NewsUsersAddReply` message fields

Name	Type	Description
result	Bool	true, if the operation to add news to a selected list is successfully performed, otherwise false

5.9.25. NewsUsersAddRequest

Request to add news to a selected list

Table 209. `NewsUsersAddRequest` message fields

Name	Type	Description
news_id	int32 (array)	List with primary record keysof a news message in IRIS IS.

5.9.26. NewsUsersDeleteReply

Reply to a request to delete news from a selected list

Table 210. `NewsUsersDeleteReply` message fields

Name	Type	Description
result	bool	true, if the operation to delete news from a selected list is successfully performed, otherwise false

5.9.27. NewsUsersDeleteRequest

Request to delete news from a selected list

Table 211. `NewsUsersDeleteRequest` message fields

Name	Type	Description
news_id	int32 (array)	List with primary record keys of a news message in IRIS IS.

5.9.28. ShortNewsInfo

Short information about a news message. It is used for broadcast.

Table 212. `ShortNewsInfo` message fields

Name	Type	Description

news_id	int32	Primary record key of a news message in IRIS IS.
news_time	DateTime	News message release time.
lang	Language	News message language.
event_type	EventType	Event type (new, update, detele).
newssections_ids	int32 (array)	List of primary keys of columns, to which a news message is linked (see Table 1, “Columns of news messages”).

5.9.29. NewsSortField

Fields under which it is possible to sort news messages.

Table 213. `Allowed values of NewsSortField`

Name	Description
NSF_NEWS_TIME	News message posting time.
NSF_TITLE	News message title.
NSF_LANG	News message language.
NSF_BEG_TIME	Event beginning date and time.
NSF_END_TIME	Event end date and time.
NSF_SOURCE_ID	Source code of news message.
NSF_VERSION	Record version.
NSF_ID	Record code in IRIS IS.

5.10. Currency instruments

5.10.1. CurOper

Basic information about currency instrument.

Table 214. `CurOper` message fields

Name	Type	Description
instrument_id	int32	Primary key of a currency instrument record in IRIS IS.
instrument_code	string	Currency instrument code.
description	string	Currency instrument description.

5.11. Currency market deals

5.11.1. CurDeal

Information about a currency market deal.

Table 215. `CurDeal` message fields

Name	Type	Description
curdeal_id	int64	Primary key of a deal record in IRIS IS.
deal_time	DateTime	Deal close time.
instrument_id	int32	Primary key of a currency instrument record in IRIS IS.
instrument_code	string	Currency instrument code.
state_id	int32	Deal status code (see Table 3, “Deal Status Code”).
session_id	int32	Cureency session code (see Table 4, “Currency Sessions Codes”).
price	Decimal	Deal price.
volume	int64	Deal volume.
swop	bool	The parameter determines whether a deal is a currency swap operation.
open_price	Decimal	Open price for currency swap.
close_price	Decimal	Close price for currency swap.
version	int64	Internal version of record in IRIS.
board	string	Trades regime code

5.11.2. CurDealsCountReply

Reply with information about count of foreign currency deals.

Table 216. `CurDealsCountReply` message fields

Name	Type	Description
count	int32	Number of counts qualified for filtration condition. Count to be returned accounts not only limitations set in filter but also limitations limitations imposed on the user executing the request.

5.11.3. CurDealsCountRequest

Request for information about count of currency market deals qualified for filtering conditions.

Note: Count to be returned accounts not only limitations fixed in filter but also limitations imposed on request performing user.

Table 217. `CurDealsCountRequest` message fields

Name	Type	Description
filter	CurDealsFilter	Filter for a requested list of currency market deals.

5.11.4. CurDealsExportReply

Reply with information about formed export file.

Table 218. `CurDealsExportReply` message fields

Name	Type	Description
url	string	URL to download generated export file.

5.11.5. CurDealsExportRequest

Request for information about currency market deals as a file in the specified format.

Table 219. `CurDealsExportRequest` message fields

Name	Type	Description
filter	CurDealsFilter	Filter for a requested list of deals.
sorting	CurDealsSortRecord (array)	Listing of fields under which sorting of a requested list of deals is required.
format	ExportFileFormat	Required format of export file.
lang	Language	Required language.

5.11.6. CurDealsFilter

Information to filter a list of currency market deals, including the requested period of time.

Table 220. `CurDealsFilter` message fields

Name	Type	Description
period	DatePeriod	Period for which a list of deals is requested.
codes	string (array)	List of currency instrument codes required for download. Both codes and templates can be transmitted for filtering. (see Section 3.22, “Filtering Templates”).
instrument_ids	int32 (array)	List of currency instrument codes required for download (see Table 6, “Currency instrument codes”).
state_ids	int32 (array)	List of required deal statuses. In case if the list is empty, all records with no limitation will be issued (see Table 3, “Codes of deals statuses”).
session_ids	int32 (array)	List of currency sessions required for download.

swop	SwopSelector	Attribute to filter currency swap
version_from	int64	Internal version of record in IRIS IS starting from which result must be issued. It is supposed that this limitation will be used in combination with indication of a bandle size (see the “bandle” field in the `CurDealsRequest` message, Section 5.11.8, “CurDealsRequest”).
curdeals_ids	int64 (array)	List of codes of deals for which it is required to obtain information. It is supposed to be used in combination with `CurDealsUpdate` (see Section 5.11.10, “CurDealsUpdate”).
confirmed	bool	If this field is filled in as `true`, only a list of confirmed deals will be issued (code 1). Otherwise, attention should be paid to a deal status transmitted in the “state_id” of the `CurDeal` message (see Table 3, “Codes of deals statuses”).

5.11.7. CurDealsReply

Reply with information about currency market deals.

Table 221. `CurDealsReply` message fields

Name	Type	Description
range	Range	Range of records in reply.
deals	CurDeal (array)	List of deals

5.11.8. CurDealsRequest

Request for information about currency market deals.

Table 222. `CurDealsRequest` message fields

Name	Type	Description
filter	CurDealsFilter	Filter for a requested list of deals.
Name	**Type**	**Description**
range	Range	Indication of a required record range. In reply to request records that match the set filter and are included in the range specified in the parameter will be transmitted. Mutually exclusive with bandle.

bandle	int32	Indication of a size of record bandles to be transmitted. In reply to request all records that match the set filter will be transmitted in packagegs of a set specified in the parameter. Mutually exclusive with range.
sorting	CurDealsSortRecord (array)	Listing of fields for which sorting of a requested list of deals is required.

5.11.9. CurDealsSortRecord

Information to sort a list of currency market deals.

Table 223. `CurDealsSortRecord` message fields

Name	Type	Description
field	CurDealsSortField	Field under which sorting is required.
desc	bool	Sorting direction. If true is indicated, reverse sorting is required.

5.11.10. CurDealsUpdate

Update on occurrence of new currency market deals or available changes.

Table 224. `CurDealsUpdate` message fields

Name	Type	Description
deals	ShortCurDealInfo (array)	Short information about deals.

5.11.11. ShortCurDealInfo

Short information about a currency market deal

Table 225. `ShortCurDealInfo` message fields

Name	Type	Description
curdeal_id	int64	Deal code.
deal_time	DateTime	Deal time.
event_type	EventType	Event type (new, update, delete).

5.11.12. CurDealsSortField

Field under which sorting of currency market deals is possible.

Table 226. `Allowed values of CurDealsSortField`

Name	Description
CDSF_DEAL_TIME	Deal conclusion timer.
CDSF_INSTRUMENT	Currency instrument code.
CDSF_PRICE	Price.

CDSF_VOLUME	Deal volume.
CDSF_VERSION	Record version.
CDSF_ID	Code of record in IRIS.

5.11.13. SwopSelector

Listed type to filter currency swap operations.

Table 227. `Allowed values of SwopSelector`

Name	Description
SS_ALL	Result must return all deals, irrespective of whether they are currency swap deals or not.
SS_YES	Result must return only currency swap deals.
SS_NO	Result must return only deals which are not currency swap.

5.12. Results of currency trades

5.12.1. CurQuotationsReply

Reply with information about currency market quotations.

For quotations in the CurTotal message, the following fields are filled in: instrument_id, instrument_code, session_id, price.open, price.high, price.low, price.close, price.close_time, price.wa, price.trend, price.trend_ps, volume.close, sum, kzt_sum, usd_sum, deal_count.

Table 228. `CurQuotationsReply` message fields

Name	Type	Description
Totals	CurTotal (array)	List of quotations.

5.12.2. CurQuotationsRequest

Request for information about currency market quotations. Each record to be returned by request will contain information about a recent instrument deal, trend and a set of data grouped for a current day which is limited as compared to CurTotalsReply.

Table 229. `CurQuotationsRequest` message fields

Name	Type	Description
filter	CurTotalsFilter	Limitation on a list of requested instruments. For a request to receive quotations, period does not have sense and will be ignored.

5.12.3. CurTotal

Information about trades on the currency market grouped by period for one instrument.

With respect to OHLC information about volume of deals (volume field), the following should be taken into account: the wa field for volume is not filled in, the low and high fields contain values not of the minimum and maximum volumes, but volume of deals at the minimum and maximum rates.

The ask_price and bid_price fields contain the best bid and ask at the time of closing the session and are calculated based on active bids at the time of closing the session. In view of this, these fields are filled in only if the information is grouped by day. For other grouping options, filling in these fields does not make sense

Table 230. `CurTotal` message fields

Name	Type	Description
instrument_id	int32	Primary record key of an instrument in IRIS IS.
instrument_code	string	Currency instrument code.
session_id	int32	Currency session code (see Table 4, “Currency Sessions Codes”).
price	Ohlc	OHLC-information for price.
volume	Ohlc	OHLC-infromation for volume of deals in currency unit.
sum	int64	Total volume of trades for a period in currency unit.
sum_trend	int64	Trend for an aggregate volume of trades for a period in currency unit against previous period.
sum_trend_ps	Decimal	Trend for an aggregate volume of trades for a period in currency unit against previous period in percentage.
kzt_sum	Decimal	Aggregate trades volume for a period in KZT.
kzt_sum_trend	Decimal	Trend for an aggregate trades volume for a period in KZT against previous period.
kzt_sum_trend_ps	Decimal	Trend for an aggregate trades volume for a period in KZT against previous period in percentage.
prev_period	TotalsPeriod	Previous period for aggregate trades volume.
deal_count	int32	Deals count for a period.
dealers_count	int32	Count of stock exchange members who participated in trades.
bid_price	Decimal	Best bid price.
ask_price	Decimal	Best ask price.
usd_sum	Decimal	Aggregate trades volume for period in USD.
period	TotalsPeriod	Period for which information is grouped (starting from version 2.10.7).

5.12.4. CurTotalsCountReply

Reply with information about count of currency market tradeds.

Table 231. `CurTotalsCountReply` message fields

Name	Type	Description
count	int32	Count of records that match the condition set in the request.

5.12.5. CurTotalsCountRequest

Request for information about count of records about currency market trades that match the filtering conditions.

Note: Count to be returned accounts not only limitations fixed in filter but also limitations imposed on request performing user.

Note: Request for a count of cannot be performed for periods which represent a range (i.e. for day_period, week_period, month_period, quarter_period and year_period).

Table 232. `CurTotalsCountRequest` message fields

Name	Type	Description
filter	CurTotalsFilter	Information to filter data about currency market trades.

5.12.6. CurTotalsExportReply

Reply with information about formed export file with information about results of currency market trades.

Table 233. `CurTotalsExportReply` message fields

Name	Type	Description
url	string	URL to download generated export file.

5.12.7. CurTotalsExportRequest

Request for information about results of currency market trades as a file in the specified format.

Table 234. `CurTotalsExportRequest` message fields

Name	Type	Description
filter	CurTotalsFilter	Information to filter data about currency market trades.
format	ExportFileFormat	Required format of export file.
lang	Language	Required language.

5.12.8. CurTotalsFilter

Information to filter data about currency market trades.

Table 235. `CurTotalsFilter` message fields

Name	Type	Description

period	TotalsPeriod	Required period to group data.
codes	string (array)	List of currency instrument codes required for download. Both codes and templates can be transmitted for filtering (see Section 3.22, “Filtering Templates”).
instrument_ids	int32 (array)	List of codes of currency instruments required for download.
session_ids	int32 (array)	List of codes of currency sessions required for download.
all_instruments	bool	The field is not used starting from version 2.11.33. To obtain similar information, CurQuotationsRequest request is added. If the field is filled in as `false` (default value), totals will contain only those instruments that match the filter and were traded in the requested period. If the field is filled as `true`, the request will return all instruments that match the filter. For those instruments that were not traded in the requested period, the last deal will be found, data on which will be filled in the `price.close`, `price.close_time`, `volume.close` (additionally to the i`nstrument_id`, `instrument_code`, `session_id` fields).

5.12.9. CurTotalsReply

Reply with information about currency market trades grouped for a period.

Table 236. `CurTotalsReply` message fields

Name	Type	Description
period	TotalsPeriod	Period for which information is grouped.
range	Range	Range of records in reply.
totals	CurTotal (array)	List of deals.

5.12.10. CurTotalsRequest

Request for information about currency market trades.

Table 237. `CurTotalsRequest` message fields

Name	Type	Description

filter	CurTotalsFilter	Information to filter data about currency market deals.
Name	**Type**	**Description**
range	Range	Indication of a required record range. In reply to request records that match the set filter and are included in the range specified in the parameter will be transmitted. Mutually exclusive with bandle.
bandle	int32	Indication of a size of record bandles to be transmitted. In reply to request all records that match the set filter will be transmitted in packagegs of a set specified in the parameter. Mutually exclusive with range.

5.12.11. CurTotalsUpdate

Update on occurrenge of grouped information about currency market trades.

Table 238. `CurTotalsUpdate` message fields

Name	Type	Description
periods	TotalsPeriod (array)	List of periods for which trades information is changed.

5.13. Currency market orders

Determination of IRIS API messages intended to transmit order information.

5.13.1. CurOrder

Structure to transmit data about a currency market order.

Table 239. `CurOrder` message fields

Name	Type	Description
curorders_id	int64	Primary record key of an order in IRIS IS.
order_time	DateTime	Order submission time.
deltime	DateTime	Order delition time.
direction	OrderDirection	Direction of a submitted order (buy, sale).
status	OrderStatus	Order status.
instrument_id	int32	Primary key of a currency instrument record in IRIS IS.
instrument_code	string	Currency instrument code.
session_id	int32	Currency session code (see Table 4, “Currency Sessions Codes”).

price	Decimal	Order price.
volume	int64	Order volume.
volrest	int64	Remaining order volume.
Name	**Type**	**Description**
version	int64	Internal version of record in IRIS.
board	string	Trades regime code

5.13.2. CurOrdersCountReply

Reply with information aboutr count of currency market orders.

Table 240. `CurOrdersCountReply` message fields

Name	Type	Description
count	int32	Count of records that match the filtering condition. Count to be returned accounts not only limitations fixed in filter but also limitations imposed on request performing user.

5.13.3. CurOrdersCountRequest

Request for information about count of currency market orders that match the filtering conditions.

Note: Count to be returned accounts not only limitations fixed in filter but also limitations imposed on request performing user.

Table 241. `CurOrdersCountRequest` message fields

Name	Type	Description
Filter	CurOrdersFilter	Filter for a requested list of currency market orders.

5.13.4. CurOrdersExportReply

Reply with information about formed export file.

Table 242. `CurOrdersExportReply` message fields

Name	Type	Description
url	string	URL to download generated export file.

5.13.5. CurOrdersExportRequest

Request for information about currency market orders as a file in the specified format.

Table 243. `CurOrdersExportRequest` message fields

Name	Type	Description
filter	CurOrdersFilter	Filter for a requested list of orders.

sorting	CurOrdersSortRecord (array)	Listing of fields for which sorting of a requested order list is required.
Name	**Type**	**Description**
format	ExportFileFormat	Required format of an export file.
lang	Language	Required language.

5.13.6. CurOrdersFilter

Information to filter st of currency market orders.

Table 244. `CurOrdersFilter` message fields

Name	Type	Description
period	DatePeriod	Period for which a list of orders is requested.
codes	string (array)	List of codes of currency instruments required for download. Both codes and templates can be transmitted for filtering (see Section 3.22, “Filtering Templates”).
instrument_ids	int32 (array)	List of codes of currency instruments required for download.
session_ids	int32 (array)	List of codes of currency sessions required for download.
direction	OrderDirection (array)	Listing of required directions of orders. If no required directions are set, directions of all directions are issued.
version_from	int64	Version starting from which records are required. It is supposed that this limitation will be used in combination with indication of a bandle size (see the “bandle” field in the `CurOrdersRequest` message, Section 5.13.8, “CurOrdersRequest”).
orders_ids	int64 (array)	List of codes of orders for which it is required to obtain information. It is supposed to be used in combination with `CurOrdersUpdate` (see Section 5.13.10, “CurOrdersUpdate”).

5.13.7. CurOrdersReply

Reply with information about currency market orders.

Table 245. `CurOrdersReply` message fields

Name	Type	Description

range	Range	Range of records in reply.
orders	CurOrder (array)	List of currency market orders.

5.13.8. CurOrdersRequest

Request for information about currency market orders.

Table 246. `CurOrdersRequest` message fields

Name	Type	Description
filter	CurOrdersFilter	Filter for a requested list of currency market orders.
range	Range	Indication of a required record range. In reply to request records that match the set filter and are included in the range specified in the parameter will be transmitted. Mutually exclusive with bandle.
bandle	int32	Indication of a size of record bandles to be transmitted. In reply to request all records that match the set filter will be transmitted in packagegs of a set specified in the parameter. Mutually exclusive with range.
sorting	CurOrdersSortRecord (array)	Listing of fields under which sorting of a requested list of currency market orders is required.

5.13.9. CurOrdersSortRecord

Information to sort a list of currency market orders.

Table 247. `CurOrdersSortRecord` message fields

Name	Type	Description
field	CurOrdersSortField	Field under which sorting is required.
desc	bool	Sorting direction. If true is indicated, reverse sorting is required.

5.13.10. CurOrdersUpdate

Update on occurrence of new currency market orders or available changes.

Table 248. `CurOrdersUpdate` message fields

Name	Type	Description
orders	ShortCurOrderInfo (array)	Short information about currency market orders.

5.13.11. ShortCurOrderInfo

Short information about order

Table 249. `ShortCurOrderInfo` message fields

Name	Type	Description
order_id	int64	Order code.
order_time	DateTime	Order time.
event_type	EventType	Event type (new, update, delete).

5.13.12. CurOrdersSortField

Fields under which it is possible to sort a list of orders.

Table 250. `Allowed values of CurOrdersSortField`

Name	Description
COSF_ORDER_TIME	Order submission time.
COSF_INSTRUMENT	Instrument.
COSF_PRICE	Price.
COSF_VOLUME	Order volume.
COSF_VERSION	Record version.
COSF_ID	Code of record in IRIS.
COSF_EXTID	External code of record (identifier used upon import from BOS DB).

5.14. Stock Market Order Book

Determination of IRIS API messages intended to transmit information about a stock market order book.

5.14.1. OrderbookReply

Reply with information about a stock market order book.

If a session is closed (session_opened = false), list of order book records must be empty. Otherwise, with an empty order book, session can be opened and the order book is empty because there are no orders.

Table 251. `OrderbookReply` message fields

Name	Type	Description
instrument_id	int32	Primary record key of an insturment in IRIS IS.
orderbook	Orderbook (array)	List of order book records.
session_opened	bool	The parameter determines session condition (open or closed).

5.14.2. OrderbookRequest

Request for information about a stock market order book.

Table 252. `OrderbookRequest` message fields

Name	Type	Description
instrument_id	int32	Primary record key of an instrument in IRIS IS.

depth	int32	Depth of order book.

5.14.3. OrderbookUpdate

Update on a change in stock market order book.

One message can arrive in response to processing of several orders (see Section 3.7, “Stock Market Order Book”).

Table 253. `OrderbookUpdate` message fields

Name	Type	Description
instrument_id	int32	Primary record key of an instrument in IRIS IS.

5.15. Currency market order book

Determination of IRIS API messages intended to transmit information about currency market order book.

5.15.1. CurOrderbookReply

Reply with information about currency market order book.

If a session is closed (session_opened = false), list of order book records must be empty. Otherwise, with an empty order book, session can be opened and the order book is empty because there are no orders.

Table 254. `CurOrderbookReply` message fields

Name	Type	Description
instrument_id	int32	Primary record key of an instrument in IRIS IS.
orderbook	Orderbook (array)	List of order book orders.
session_opened	Bool	The parameter determines session condition (open or closed).

5.15.2. CurOrderbookRequest

Request for information about currency market order book.

Table 255. `CurOrderbookRequest` message fields

Name	Type	Description
instrument_id	int32	Primary record key of an instrument in IRIS IS.
depth	int32	Depth of order book.

5.15.3. CurOrderbookUpdate

Update on a change in the currency market order book.

One message can come in reply to processing of several orders at once (see Section 3.12, “Currency Market Order Book”).

Table 256. `CurOrderbookUpdate` message fields

Name	Type	Description

instrument_id	int32	Primary record key of an instrument in IRIS IS.

5.16. Stock market graphs

5.16.1. GraphReply

Reply with information to plot graphs.

Table 257. `GraphReply` message fields

Name	Type	Description
security_id	int32	Primary key of security record in IRIS IS.
period	TotalsPeriod	Period for which data are provided.
interval	GraphIntervalType	Type of interval between counting in series.
ohlc	GraphOhlc (array)	List of OHLC values.

5.16.2. GraphRequest

Request for information to plot stock market graphs.

Table 258. `GraphRequest` message fields

Name	Type	Description
security_id	int32	Primary key of security record in IRIS IS. The field must be filled in.
period	TotalsPeriod	Period for which data are requested. If no period is set, in reply to request graph for the whole time of trades on the stock market will be issued (starting from September 1997).
need_empty_intervals	bool	It determines whether empty intervals, for example intervals throughout which no deal was made, should be included in the result.

5.16.3. HeatmapRecord

Information about one security in data for a heat map.

Table 259. `HeatmapRecord` message fields

Name	Type	Description
security_id	int32	Primary key of security record in IRIS IS.
code	string	Security code.

Name	Type	Description
capitalization	Decimal	Issuer capitalization in KZT. Value of cell of the heat map depends on capitalization.
trend	Decimal	Trend in % - is a ratio of price the last successful session to the last price of a previous session. Color of a cell of the heat map depends on a trend color.
last_price	Decimal	Last deal price.
last_time	DateTime	Last deal conclusion time.

5.16.4. HeatmapReply

Reply with information to plot a heat map.

Table 260. `HeatmapReply` message fields

Name	Type	Description
type	HeatmapType	Requested variat of heat map.
records	HeatmapRecord (array)	List of securities in heat map.

5.16.5. HeatmapRequest

Request for data to plot the heat map.

Table 261. `HeatmapRequest` message fields

Name	Type	Description
type	HeatmapType	Required variant of the heat map.
depth	HeatmapDepth	Required calculation depth of trend. If no parameter is specified, the trend is calculated to the last price of a previous successful session (HD_DAY).

5.16.6. HeatmapDepth

Required trend depth upon calculation of the heat map.

Table 262. `Allowed values of HeatmapDepth`

Name	Description
HD_DAY	Day.
HD_MONTH	Month.
HD_QUARTER	Quarter.
HD_YEAR	Year.

5.16.7. HeatmapType

Listed type to specify the required variant of the heat map.

Table 263. `Allowed values of HeatmapType`

Name	Description

HT_INDEX_KASE	Shares of KASE index.
HT_KASE_GLOBAL	Foreign shares of KASE Global platform.

5.17. Currency Market Graphs

5.17.1. CurGraphReply

Reply with information to plot graphs.

Table 264. `CurGraphReply` message fields

Name	Type	Description
security_id	int32	Primary key of a currency instrument record in IRIS IS.
period	TotalsPeriod	Period for which data are provided.
interval	GraphIntervalType	Type of interval between counting in series.
ohlc	GraphOhlc (array)	List of OHLC values.

5.17.2. CurGraphRequest

Request for information to plot currency market graphs.

Table 265. `CurGraphRequest` message fields

Name	Type	Description
instrument_id	int32	Primary key of a currency instrument record in IRIS IS. The field must be filled in.
period	TotalsPeriod	Period for which data are requested. If no period is set in reply to request graph for the entire period of currency market trades will be issued (starting from December 1998).
need_empty_intervals	bool	It determines whether empty intervals, for example intervals throughout which no deal was made, should be included in the result.

5.18. Indicators

5.18.1. IndChangeReply

Reply with information about change in value of the selected inducator one year before the specified date.

Table 266. `IndChangeReply` message fields

Name	Type	Description

type_id	int32	Primary record key of indicator type in IRIS IS (see Table 7, “Codes of Indicators Types”).
last_time	DateTime	Time of the last calculation of indicator value on a date on which information was requested.
last_value	Decimal	Last indicator value on a date on which information was requested.
Name	**Type**	**Description**
day_value	Decimal	Indicator value one day before request date.
week_value	Decimal	Indicator value one week before request date.
month_value	Decimal	Indicator value one month before request date.
quarter_value	Decimal	Value indicator one quarter before request date.
half_year_value	Decimal	Value indicator six months before request date.
year_value	Decimal	Value indicator one year before request date.

5.18.2. IndChangeRequest

Request for information about change in value of a selected indicator one year before the specified date.

Table 267. `IndChangeRequest` message fields

Name	Type	Description
type_id	int32	Primary record key of an indicator type in IRIS IS (see Table 7, “Codes of Indicators Types”).
request_date	Date	Date on which information is requested. If no parameter is set, information on a current date will be issued.

5.18.3. IndGraphReply

Reply with information to plot indicator graphs.

Table 268. `IndGraphReply` message fields

Name	Type	Description
type_id	int32	Primary record key of an indicator type in IRIS IS (see Table 7, “Codes of Indicators Types”).
period	TotalsPeriod	Period for which data are provided.
interval	GraphIntervalType	Type of interval between counting in series.

ohlc	GraphOhlc (array)	List of OHLC values.

5.18.4. IndGraphRequest

Request for information to plot indicator graphs.

Table 269. `IndGraphRequest` message fields

Name	Type	Description
type_id	int32	Primary record key of an indicator type in IRIS IS (see Table 7, “Codes of Indicators Types”).
Name	**Type**	**Description**
		The field must be filled in.
period	TotalsPeriod	Period for which data are requested. If no period is set, in reply to request graph for the entire period of currency market trades will be issued (starting from November 1993).
need_empty_intervals	bool	It determines whether empty intervals, for example intervals throughout which no indicator value was calculated, should be included in the result.

5.18.5. IndQuotesReply

Reply with a list of recent values of indicators.

Table 270. `IndQuotesReply` message fields

Name	Type	Description
indicators	Indicator (array)	List of recent values of indicators.

5.18.6. IndQuotesRequest

Request for a list of recent values of indicators.

Table 271. `IndQuotesRequest` message fields

Name	Type	Description
type_ids	int32 (array)	List of codes of indicators types required for download (see Table 7, “Codes of Indicators Types”). In case of an empty list, an error message will be received.

5.18.7. Indicator

Information about value of a stock market indicator.

Table 272. `Indicator` message fields

Name	Type	Description

indicator_id	int32	Primary record key of an indicator value in IRIS IS.
value_time	DateTime	Calculation time of indicator value.
type_id	int32	Primary record key of an indicator type in IRIS IS (see Table 7, “Codes of Indicators Types”).
value	Decimal	Indicator value. For IndQuotesRequest request, it is the last indicator value.
Name	**Type**	**Description**
version	int64	Internal version of record in IRIS.
trend_value	Decimal	Previous indicator value.
trend_time	DateTime	Calculation time of indicator value for trend.
day_open	Decimal	Indicator value at the beginning of the last settelemtn date. It has sense for real time indicators.
day_min	Decimal	Minimum indicator value during the last settlement date. It has sense for real time indicators.
day_max	Decimal	Maximum indicator value during the last settlement date. It has sense for real time indicators.
year_min	Decimal	Minimum indicator value for the last 52 weeks (including current settlement date).
year_max	Decimal	Maximum indicator value for the last 52 weeks (including current settlement date).
min	Decimal	Historic minimum of indicator value (including current settlement date).
max	Decimal	Historic maximum of indicator value (including current settlement date).

5.18.8. IndicatorInGroup

Minimum information about indicator.

Table 273. `IndicatorInGroup` message fields

Name	Type	Description
type_id	int32	Primary record key of an indicator type in IRIS IS (see Table 7, “Codes of Indicators Types”).

name	string	Indicator name. The field contains indicator name which is actual at the time of request. Names in reference table (see Table 7, “Codes of Indicators Types”) are provided for information only.
realtime	bool	Frequency of indicator calculation. In case of true, indicator is calculated in real time. In case of false, calculation is made once a day.

5.18.9. IndicatorsCountReply

Reply with information about count of records of indicator values which match the filtering condition.

Table 274. `IndicatorsCountReply` message fields

Name	Type	Description
count	int32	Count of records that match the filtering condition. Count to be returned accounts not only limitations fixed in filter but also limitations imposed on request performing user.

5.18.10. IndicatorsCountRequest

Request for information about count of records of indicator values which match the filtering condition.

Note: Count to be returned accounts not only limitations fixed in filter but also limitations imposed on request performing user.

Table 275. `IndicatorsCountRequest` message fields

Name	Type	Description
filter	IndicatorsFilter	Filter for a requested list of indicator values.

5.18.11. IndicatorsExportReply

Reply with information about formed export file.

Table 276. `IndicatorsExportReply` message fields

Name	Type	Description
url	string	URL to download generated export file.

5.18.12. IndicatorsExportRequest

Request for information about list of indicators as a file in the specified format.

Table 277. `IndicatorsExportRequest` message fields

Name	Type	Description
filter	IndicatorsFilter	Filter for a requested list of indicators.

sorting	IndicatorsSortRecord (array)	Listing of the fields under which sorting of a requested indicator list is required.
format	ExportFileFormat	Required format of export file.
lang	Language	Required language.

5.18.13. IndicatorsFilter

Information to filter a list of values of stock market indicators, including the requeswted time period.

Table 278. `IndicatorsFilter` message fields

Name	Type	Description
period	DatePeriod	Period for which a list of indicator values is requested.
type_ids	int32 (array)	List of codes of indicators types required for download (see Table 7, “Codes of Indicators Types”).
version_from	int64	Internal version of record in IRIS IS starting from which result must be issued. It is supposed that this limitation will be used in combination with indication of a bandle size (see the “bandle” field in the `IndicatorsRequest` message, Section 5.18.16, “IndicatorsRequest”).
indicator_ids	int32 (array)	List of deals codes for which it is required to obtain information. It is supposed to be used in combination with `IndicatorsUpdate` (see Section 5.18.18, “IndicatorsUpdate”).

5.18.14. IndicatorsGroup

Information about group of indicators.

Table 279. `IndicatorsGroup` message fields

Name	Type	Description
name	string	Name of group of indicators.
indicators	IndicatorInGroup (array)	List of indicators included in group. Breakdown of indicators into groups in reference table is set out for information only.

5.18.15. IndicatorsReply

Reply with a list of values of indicators that match the filtering condition.

In reply obtained in response to the IndicatorsRequest request the following fields are not filled in: day_open, day_min, day_max, year_min, year_max, min and max.

Table 280. `IndicatorsReply` message fields

Name	Type	Description
range	Range	Range of records in reply.
indicators	Indicator (array)	List of indicator values.

5.18.16. IndicatorsRequest

Request for a list of indicator values that match the filtering condition.

Table 281. `IndicatorsRequest` message fields

Name	Type	Description
filter	IndicatorsFilter	Filter for a requested list of indicator values.
Name	**Type**	**Description**
range	Range	Indication of a required record range. In reply to request records that match the set filter and are included in the range specified in the parameter will be transmitted. Mutually exclusive with bandle.
bandle	int32	Indication of a size of record bandles to be transmitted. In reply to request all records that match the set filter will be transmitted in packagegs of a set specified in the parameter. Mutually exclusive with range.
sorting	IndicatorsSortRecord (array)	Listing of fields under which sorting of a requested list of deals is required.

5.18.17. IndicatorsSortRecord

Information to sort a list of values of stock market indicators.

Table 282. `IndicatorsSortRecord` message fields

Name	Type	Description
field	IndicatorsSortField	Field under which sorting is required.
desc	bool	Sorting direction. If true is indicated, reverse sorting is required.

5.18.18. IndicatorsUpdate

Update on occurrence of new values of indicators or availability of changes.

Table 283. `IndicatorsUpdate` message fields

Name	Type	Description
indicators	ShortIndicatorInfo (array)	Short information about indicator values.

5.18.19. ShortIndicatorInfo

Short information about value of a stock market indicator.

Table 284. `ShortIndicatorInfo` message fields

Name	Type	Description
indicator_id	int32	Primary record key of an indicator value in IRIS IS.
value_time	DateTime	Calculation time of indicator value.
type_id	int32	Primary record key of an indicator type in IRIS IS (see Table 7, “Codes of Indicators Types”).
event_type	EventType	Event type (new, update, delete).

5.18.20. IndicatorsSortField

Fields under which a list of indicator values can be sorted.

Table 285. `Allowed values of IndicatorsSortField`

Name	Description
ISF_VALUE_TIME	Calculation time of indicator value.
ISF_TYPE	Primary record key of an indicator type in IRIS IS.
ISF_VALUE	Indicator value.
ISF_VERSION	Record version.
ISF_ID	Code of record in IRIS.

5.19. Repo market deals

5.19.1. Repo

Information about deal on the repo market.

It should be noted that the `auto_id` and `auto_code` fields are filled in only for historical data. In the new MOEX trading system, the concept of an automatic repo instrument is absent and the auto_code field will always be filled with the value "MOEX".

The `prolong_date` field is filled in only if the repo operation was extended.

The `close_vol_kzt` and `close_vol_usd` fields are filled in only after the repo operation is closed.

Table 286. `Repo` message fields

Name	Type	Description
repo_id	int32	Primary record key of a repo deal in IRIS IS.
open_time	DateTime	Repo operation open time.
close_time	DateTime	Repo operation close time.
prolong_date	Date	Repo operation prolongation time.

kred_days	int32	Repo term in days.
sec_id	int32	Primary key of security record in IRIS IS.
sec_code	string	Automatic repo instrument code.
autorepo	bool	Automatic repo deal sign.
auto_id	int32	Primary record key of an automatic repo deal in IRIS IS.
auto_code	string	Security code.
open_price	Decimal	Repo operation open price.
close_price	Decimal	Repo operation close price.
volume	int64	Repo operation volume.
open_vol_kzt	Decimal	Repo operation open volume in KZT.
open_vol_usd	Decimal	Repo operation open volume in USD at the exchange rate accepted on “KASE” on the operation open day.
Name	**Type**	**Description**
close_vol_kzt	Decimal	Repo operation close volume in KZT.
close_vol_usd	Decimal	Repo operation close volume in USD at the exchange rate accepted on “KASE” on the operation close day.
yield	Decimal	Repo operation yield.
version	int64	Internal version of record in IRIS.
board	string	Trades regime code
non_kase	bool	Sign of a repo deal transacted outside KASE.
sec_type	SecType	Security type.
price_type	PriceType	Price type.
actual_sin	string	Security identification (NIN, ISIN, etc.) valid on the deal date.

5.19.2. RepoCountReply

Reply with information about count of repo market deals.

Table 287. `RepoCountReply` message fields

Name	Type	Description

count	int32	Count of records that match the filtering condition. Count to be returned accounts not only limitations fixed in filter but also limitations imposed on request performing user.

5.19.3. RepoCountRequest

Request for information about count of repo market deals that match the filtering conditions.

Note: Count to be returned accounts not only limitations fixed in filter but also limitations imposed on request performing user.

Table 288. `RepoCountRequest` message fields

Name	Type	Description
filter	RepoFilter	Filter for a requested list of repo market deals.

5.19.4. RepoExportReply

Reply with information about formed export file.

Table 289. `RepoExportReply` message fields

Name	Type	Description
url	string	URL to download generated export file.

5.19.5. RepoExportRequest

Request for information about repo market deals as a file in the specified format.

Table 290. `RepoExportRequest` message fields

Name	Type	Description
filter	RepoFilter	Filter for a requested list of deals.
sorting	RepoSortRecord (array)	Listing of fields under which sorting of a requested list of deals is required.
format	ExportFileFormat	Required format of export file.
lang	Language	Required language.

5.19.6. RepoFilter

Information to filter a list of repo market deals.

Table 291. `RepoFilter` message fields

Name	Type	Description
period	DatePeriod	Period for which a list of deals is requested.

instruments_filter	InstrumentsFilter	Filter to fix limitations in the breakdown of instruments with which deals were made.
version_from	int64	Version starting from which records are required. It is supposed that this limitation will be used in combination with indication of a bandle size (see the “bandle” field in the `RepoRequest` message, Section 5.19.9, “RepoRequest”).
repo_ids	int64 (array)	List of repo deal codes for which it is required to obtain information. It is supposed to be used in combination with `RepoUpdate` (see Section 5.19.11, “RepoUpdate”).

5.19.7. RepoQuote

Information about quotation on the repo market. It is used in replyes to requests under a user watchlist.

Table 292. `RepoQuote` message fields

Name	Type	Description
instrument_id	int32	Primary record key of an instrument in IRIS IS.
board	string	Trades regime code.
code	string	Repo deal code. It is an instrument code combined with description of a Trades regime translated in a Request language.
yield	Decimal	Yield of the last open deal of repo operation.
last_time	DateTime	Time of the last open deal of repo operation.
yield_min	Decimal	Minimum yield of open deals during the last trading day.
yield_max	Decimal	Maximum yield of open deals during the last trading day.
volume_sum	int64	Aggregate volume of open deals during the last trading day.

5.19.8. RepoReply

Reply with information about repo market deals.

Table 293. `RepoReply` message fields

Name	Type	Description
range	Range	Range of records in reply.
deals	Repo (array)	List of repo market deals.

5.19.9. RepoRequest

Request for information about repo market deals.

Table 294. `RepoRequest` message fields

Name	Type	Description
filter	RepoFilter	Filter for a requested list of repo market deals.
range	Range	Indication of a required record range. In reply to request records that match the set filter and are included in the range specified in the parameter will be transmitted. Mutually exclusive with bandle.
bandle	int32	Indication of a size of record bandles to be transmitted. In reply to request all records that match the set filter will be transmitted in packagegs of a set specified in the parameter. Mutually exclusive with range.
sorting	RepoSortRecord (array)	Listing of fields under which sorting of a requested list of deals is required.

5.19.10. RepoSortRecord

Information to sort a list of repo market deals.

Table 295. `RepoSortRecord` message fields

Name	Type	Description
field	RepoSortField	Field under which sorting is required.
desc	bool	Sorting direction. If true is indicated, reverse sorting is required.

5.19.11. RepoUpdate

Update on occurrence of new repo market deals or available changes.

Table 296. `RepoUpdate` message fields

Name	Type	Description
deals	ShortRepoInfo (array)	Short information about repo market deals.

5.19.12. ShortRepoInfo

Short information about a repo deal

Table 297. `ShortRepoInfo` message fields

Name	Type	Description

repo_id	int32	Primary record key of a repo deal in IRIS IS.
open_time	DateTime	Repo deal open time.
close_time	DateTime	Repo deal close time.
event_type	EventType	Event type (new, update, delete).

5.19.13. RepoSortField

Fields under which it is possible to sort a list on the repo market.

Table 298. `Allowed values of RepoSortField`

Name	Description
RSF_OPEN_TIME	Repo operation open time.
RSF_CLOSE_TIME	Repo operation close time.
RSF_PROLONG_DATE	Repo operation prolongation time.
RSF_KRED_DAYS	Repo term in days.
RSF_SEC_CODE	Security code.
RSF_AUTO_CODE	Automatic repo instrument code.
RSF_OPEN_PRICE	Repo operation open price.
RSF_CLOSE_PRICE	Repo operation close price.
RSF_VOLUME	Repo operation volume.
RSF_OPEN_VOL_KZT	Open volme of repo operation in KZT.
RSF_OPEN_VOL_USD	Open volme of repo operation in USD.
RSF_CLOSE_VOL_KZT	Close volme of repo operation in KZT.
RSF_CLOSE_VOL_USD	Close volme of repo operation in USD.
RSF_YIELD	Yield.
RSF_VERSION	Record version.
RSF_ID	Primary record key of repo deal in IRIS IS.
RSF_EXTID	External code of record (identifier used upon import from BOS DB).

5.20. Repo market orders

Determination of IRIS API messages intended to transmit information about repo orders.

5.20.1. RepoOrder

Structure for transfer of data about a repo request.

Table 299. `RepoOrder` **message fields**

Name	Type	Description
repoorders_id	int64	Primary record key of an order in IRIS IS.

serial	int64	Order number in trading system
order_time	DateTime	Order submission time.
deltime	DateTime	Order deletion time.
direction	OrderDirection	Direction of a submitted order (buy, sale).
status	OrderStatus	Order status.
instrument_id	int32	Primary record key of an instrument in IRIS IS.
instrument_code	string	Instrument code.
actual_sin	string	Valid identification number of a security (NIN, ISI, etc.)
currencies_id	int32	Primary code of currency of a submitted order in IRIS IS (see Table 5, “Currency Codes”).
Name	**Type**	**Description**
price	Decimal	Order price.
yield	Decimal	Yield.
volume	int64	Order volume.
volrest	int64	Remaining order volume.
version	int64	Internal version of a record in IRIS.
board	string	Trades regime code
vol_kzt	Decimal	Order volume in KZT.
vol_usd	Decimal	Order volume in USD at the exchange rate acceted at “KASE” on a date of order submission.
volcontr	Decimal	Order volume in USD at the exchange rate acceted at “KASE” on a date of order submission.
sec_type	SecType	Security type.

5.20.2. RepoOrdersCountReply

Reply with information about a count of repo orders.

Table 300. `RepoOrdersCountReply` message fields

Name	Type	Description
count	int32	Count of records that match the filtering condition. Count to be returned accounts not only limitations fixed in filter but also limitations imposed on request performing user.

5.20.3. RepoOrdersCountRequest

Request for information about a count of repo orders that match the filtering conditions.

Note: Count to be returned accounts not only limitations fixed in filter but also limitations imposed on request performing user.

Table 301. `RepoOrdersCountRequest` message fields

Name	Type	Description
filter	RepoOrdersFilter	Filter for a requested list of repo orders.

5.20.4. RepoOrdersExportReply

Reply with information about formed export file.

Table 302. `RepoOrdersExportReply` message fields

Name	Type	Description
url	string	URL to download generated export file.

5.20.5. RepoOrdersExportRequest

Request for information repo orders as a file in the specified format.

Table 303. `RepoOrdersExportRequest` message fields

Name	Type	Description
filter	RepoOrdersFilter	Filter for a requested list of orders.
sorting	RepoOrdersSortRecord (array)	Listing of fields for which sorting of a requested order list is required.
format	ExportFileFormat	Required format of export file.
lang	Language	Required language.

5.20.6. RepoOrdersFilter

Information to filter a list of repo orders.

Table 304. `RepoOrdersFilter` message fields

Name	Type	Description
period	DatePeriod	Period for which a list of orders is requested.
codes	string (array)	List of instrument codes required for download. Both codes and templates can be transmitted for filtering (see Section 3.22, “Filtering Templates”).
instrument_ids	int32 (array)	List of instrument codes required for download.

direction	OrderDirection (array)	Listing of required order directions. If no required directions are set, orders of all directors are issued.
version_from	int64	Version starting from which records are required. It is supposed that this limitation will be used in combination with indication of a bandle size (see the “bandle” field in the `OrdersRequest` message, Section 5.5.8, “OrdersRequest”).
repoorders_ids	int64 (array)	List of order codes for which it is required to obtain information. It is supposed to be used in combination with `OrdersUpdate` (see Section 5.5.10, “OrdersUpdate”).

5.20.7. RepoOrdersReply

Reply with information about repo orders.

Table 305. `RepoOrdersReply` message fields

Name	Type	Description
range	Range	Range of records in reply.
repoorders	RepoOrder (array)	List of repo orders.

5.20.8. RepoOrdersRequest

Request for information about repo orders.

Table 306. `RepoOrdersRequest` message fields

Name	Type	Description
filter	RepoOrdersFilter	Filter for a requested list of repo orders.
range	Range	Indication of a required record range. In reply to request records that match the set filter and are included in the range specified in the parameter will be transmitted. Mutually exclusive with bandle.
bandle	int32	Indication of a size of record bandles to be transmitted. In reply to request all records that match the set filter will be transmitted in packagegs of a set specified in the parameter. Mutually exclusive with range.
sorting	RepoOrdersSortRecord (array)	Listing of fields under which it is required to sort a requested list of repo orders.

5.20.9. RepoOrdersSortRecord

Information to sort a list of repo orders.

Table 307. `RepoOrdersSortRecord` message fields

Name	Type	Description
field	RepoOrdersSortField	Field under which sorting is required.
desc	bool	Sorting direction. If true is indicated, reverse sorting is required.

5.20.10. RepoOrdersUpdate

Update on occurrece of new repo orders or available changes.

Table 308. `RepoOrdersUpdate` message fields

Name	Type	Description
repoorders	ShortRepoOrderInfo (array)	Short information about repo orders.

5.20.11. ShortRepoOrderInfo

Short information about a repo order

Table 309. `ShortRepoOrderInfo` message fields

Name	Type	Description
repoorder_id	int64	Order code.
Name	**Type**	**Description**
order_time	DateTime	Order time
event_type	EventType	Event type (new, update, delete).

5.20.12. RepoOrdersSortField

Fields under which it is possible to sort a list of repo orders.

Table 310. `Allowed values of RepoOrdersSortField`

Name	Description
ROSF_ORDER_TIME	Order submission time.
ROSF_INSTRUMENT	Instrument.
ROSF_PRICE	Price.
ROSF_VOLUME	Order volume.
ROSF_YIELD	Yield.
ROSF_VERSION	Record version.
ROSF_ID	Code of record in IRIS.
ROSF_EXTID	External code of record (identifier used upon import from BOS DB).

5.21. Repo market order book

Determination of IRIS API messages designed to transmit information about a repo market order book.

5.21.1. RepoOrderbookReply

Reply with information about repo market order book.

If a session is closed (session_opened = false), list of order book records must be empty. Otherwise, with an empty order book, session can be opened and the order book is empty because there are no orders.

Table 311. `RepoOrderbookReply` message fields

Name	Type	Description
instrument_id	int32	Primary record key of an instrument in IRIS IS.
board	string	Trades regime code.
orderbook	Orderbook (array)	List of order book orders.
session_opened	bool	The parameter determines session condition (open or closed).

5.21.2. RepoOrderbookRequest

Request for information about a repo market order book.

Table 312. `RepoOrderbookRequest` message fields

Name	Type	Description
instrument_id	int32	Primary record key of an instrument in IRIS IS.
Name	**Type**	**Description**
board	string	Trades regime code.
depth	int32	Depth of order book.

5.21.3. RepoOrderbookUpdate

Notice of a change in a repo market order book.

One message can come in response to processing of several orders (see Section 3.7, “Stock Market Order Book”).

Table 313. `RepoOrderbookUpdate` message fields

Name	Type	Description
instrument_id	int32	Primary record key of an instrument in IRIS IS.
board	string	Trades regime code.

5.22. User Watchlist

5.22.1. WatchlistAddReply

Reply with information about results of adding the instruments to a user watchlist.

Table 314. `WatchlistAddReply` message fields

Name	Type	Description
ok	bool	Status: `true` if the additing to the list has been successful.

5.22.2. WatchlistAddRequest

Request to add instruments to a user watchlist.

Table 315. `WatchlistAddRequest` message fields

Name	Type	Description
instruments	WatchlistInstrument (array)	List of instruments and indicators to be added to the list.

5.22.3. WatchlistDelReply

Reply with information about results of deleting the instruments from the user watchlist.

Table 316. `WatchlistDelReply` message fields

Name	Type	Description
ok	bool	Staus: `true` if the deleting from the list has been successful.

5.22.4. WatchlistDelRequest

Request to delete instruments from the user watchlist.

Table 317. `WatchlistDelRequest` message fields

Name	Type	Description
instruments	WatchlistInstrument (array)	List of instruments and indiators to be removed from the list.

5.22.5. WatchlistInstrument

Instrument in a user watchlist.

Table 318. `WatchlistInstrument` message fields

Name	Type	Description
instrument_id	int32	Primary instrument key in IRIS IS.
type	WatchlistInstrumentType	Type of instrument code of which is set out in the instrument_id field (stock, currency or indciator).
code	string	Instrument code. The field is filled in search results returned by server. In list add and delete requests filling into this field is not compulsory.

board	string	Trades regime code. The field is used in search results for repo market instruments. It is compulsory in list add and delete requests for repo instruments.

5.22.6. WatchlistReply

Reply with information about quotes for instruments included in a user watchlist.

For quotation, in the Total message the following fields are filled in: instrument_id, instrument_code, sec_type, price.open, price.high, price.low, price.close, price.close_time, price.wa, price.trend, price.trend_ps, volume.close, kzt_sum, usd_sum, deal_count.

For quotations in the CurTotal message, the following fields are filled in: instrument_id, instrument_code, deal_count, sum, price.high, price.low, price.close, price.close_time.

Table 319. `WatchlistReply` message fields

Name	Type	Description
sec_quotes	Total (array)	List of stock market quotations.
cur_quotes	CurTotal (array)	List of currency market quotations.
ind_quotes	Indicator (array)	List of recent values of indicators.
repo_quotes	RepoQuote (array)	List of repo market quotations.

5.22.7. WatchlistRequest

Request to receive quotations for instruments included in a user watchlist.

Table 320. `WatchlistRequest` message fields

Name	Type	Description
types	WatchlistInstrumentType (array)	Optionally. Types of instruments for which it is required too issue quotations. In case if the field is not filled in, quotations will be returned for all instrument types which the user added to the watchlist.
lang	Language	Required language for the watchlist. Currently, it is related only to the code field in the RepoQuote message which is an instrument code combied with description of the trades regime. Where the field is not filled in, result will be in the Russian language in order to ensure reverse compatibility.

5.22.8. WatchlistSearchReply

Reply with information about instruments upon adding to a user watchlist.

Table 321. `WatchlistSearchReply` message fields

Name	Type	Description
instruments	WatchlistInstrument (array)	A list of instruments and indicators that match a search condition.

5.22.9. WatchlistSearchRequest

Request to search for instruments upon adding to a user watchlist.

Not more than 100 instruments of each type are returned in a reply to request.

Table 322. `WatchlistSearchRequest` message fields

Name	Type	Description
code	string	Code of instrument and indicator required for download. Both codes and templates can be transmitted for filtering (see Section 3.22, “Filtering Templates”). Length of code transmited in request must not be less than 3 symbols.
lang	Language	Required language for search.

5.22.10. WatchlistUpdate

Update of users the watchlists of which are subject to update.

Users must update a watchlist in two cases: if they received the WatchlistUpdate message with an empty name list in the parameter user_names or if their name is in user_names.

Table 323. `WatchlistUpdate` message fields

Name	Type	Description
user_names	string (array)	List of user names the watchlists of which should be updated.

5.22.11. WatchlistInstrumentType

Instrument type in a user watchlist.

Table 324. `Allowed values of WatchlistInstrumentType`

Name	Description
WIT_SECURITIES	Stock market.
WIT_CURRENCIES	Currency market.
WIT_INDICATORS	Indicators.
WIT_REPO	Repo operations.

5.23. Open information

Determination of IRIS API messages designed to transmit general open informaiton.

5.23.1. RegError

Message about a wrong filling of a field ipon registration.

Table 325. `RegError` message fields

Name	Type	Description
id	String	Identifier of a field that was subject to a check (see Table 8, “Identifier of field which was subject to check upon user registration”).
message	String	Error message upon field check.

5.23.2. ResetPwdReply

Reply with a result of field reset.

Table 326. `ResetPwdReply` message fields

Name	Type	Description
ok	bool	Stats: `true` where password reset was successful. Otherwise the **message** field will contain an error message.
message	string	Field with an error message.

5.23.3. ResetPwdRequest

Request to reset a user password.

Password is changed to a new one created in random manner and is sent to a user e-mail.

Table 327. `ResetPwdRequest` message fields

Name	Type	Description
lang	Language	Language of error messages.
user_name	string	Name of user for which it is necessary to reset password.

5.23.4. User

Information for user registration.

Table 328. `User` message fields

Name	Type	Description
firstname	string	Name (maximum length is 24 symbols).
lastname	string	Last name (maximum length is 24 symbols).
middlename	string	Patronymic name (maximum length is 24 symbols).
email	string	Email address (maximum length is 100 symbols).

user_name	string	User name (maximum length is 24 symbols). Where no user name is set or it is duplicated, it will be formed on server and sent in a registration letter.
password	string	Password (maximum length is 20 symbols). Where no password is set or it does not meet rules, it will be formed on server and sent in a registration letter.
tel	string	Telephone
address	string	Address
comments	string	Comments

5.23.5. UserRegReply

Reply with user registration result.

Table 329. `UserRegReply` message fields

Name	Type	Description
ok	bool	Registration status. `true` where registration is successful. Otherwise, the **messages** field will contain a list of error messages.
messages	RegError (array)	List of error messages upon check of fields of a registration request.

5.23.6. UserRegRequest

User registration request.

Table 330. `UserRegRequest` message fields

Name	Type	Description
user	User	User information for registration.
lang	Language	User preferred language.
guid	string	Unique identifier
reCaptcha	string	Captcha line.

5.24. Basic scalar type

Type	Note	C++	Java	Python
double		double	double	float
float		float	float	float
int32	Uses variable-length encoding. Inefficient for encoding negative numbers – if your field is likely to have negative values, use sint32 instead.	int32	int	int

int64	Uses variable-length encoding. Inefficient for encoding negative numbers – if your field is likely to have negative values, use sint64 instead.	int64	long	int/long
uint32	Uses variable-length encoding.	uint32	int	int/long
uint64	Uses variable-length encoding.	uint64	long	int/long
sint32	Uses variable-length encoding. Signed int value. These more efficiently encode negative numbers than regular int32s.	int32	int	int
sint64	Uses variable-length encoding. Signed int value. These more efficiently encode negative numbers than regular int64s.	int64	long	int/long
fixed32	Always four bytes. More efficient than uint32 if values are often greater than 2^28.	uint32	int	int
fixed64	Always eight bytes. More efficient than uint64 if values are often greater than 2^56.	uint64	long	int/long
sfixed32	Always four bytes.	int32	int	int
sfixed64	Always eight bytes.	int64	long	int/long
bool		bool	boolean	boolean
string	A string must always contain UTF-8 encoded or 7bit ASCII text.	string	String	str/unicode
bytes	May contain any arbitrary sequence of bytes.	string	ByteString	str