

Yamcs HTTP API

Release 5.7.3-SNAPSHOT

Space Applications Services, NV/SA

Leuvensesteenweg 325
1932 Sint-Stevens-Woluwe
Belgium
spaceapplications.com
yamcs.org

Aerospace Applications North America, Inc.

16850 Saturn Ln, Ste 100
Houston, TX 77058
United States of America
aerospaceapplications-na.com

Contents

1	API Overview	1
2	WebSocket	3
2.1	Connection	3
2.2	Client Message	3
2.2.1	Built-in Client messages	4
2.3	Server Messages	4
2.3.1	Built-in Server messages	5
2.4	Example	5
3	Alarms	7
3.1	List Alarms	7
3.2	List Processor Alarms	17
3.3	Edit Alarm	26
3.4	Acknowledge Alarm	27
3.5	Shelve Alarm	27
3.6	Unshelve Alarm	28
3.7	Clear Alarm	28
3.8	Subscribe Global Status	29
3.9	Subscribe Alarms	29
4	Audit	40
4.1	List Audit Records	40
5	Buckets	42
5.1	List Buckets	42
5.2	Get Bucket	43
5.3	Create Bucket	44
5.4	Delete Bucket	44
5.5	Get Object	44
5.6	Upload Object	45
5.7	List Objects	46
5.8	Delete Object	47
6	Clearance	48
6.1	List Clearances	48
6.2	Update Clearance	48
6.3	Delete Clearance	49
6.4	Subscribe Clearance	49
7	Commands	51
7.1	Issue Command	51
7.2	Update Command History	53
7.3	List Commands	54
7.4	Get Command	57
7.5	Stream Commands	58
7.6	Subscribe Commands	60

7.7	Export Command	61
8	Cop1	63
8.1	Initialize	63
8.2	Resume	64
8.3	Disable	64
8.4	Update Config	64
8.5	Get Config	65
8.6	Get Status	66
8.7	Subscribe Status	67
9	Database	70
9.1	List Databases	70
9.2	Get Database	70
10	Events	72
10.1	List Events	72
10.2	Create Event	73
10.3	List Event Sources	75
10.4	Stream Events	75
10.5	Export Events	76
10.6	Subscribe Events	77
11	File Transfer	79
11.1	List File Transfer Services	79
11.2	List Transfers	80
11.3	Get Transfer	81
11.4	Create Transfer	82
11.5	Pause Transfer	84
11.6	Cancel Transfer	84
11.7	Resume Transfer	84
11.8	Subscribe Transfers	85
12	lam	87
12.1	List Privileges	87
12.2	List Roles	87
12.3	Get Role	88
12.4	Delete Role Assignment	89
12.5	List Users	89
12.6	Get User	91
12.7	Create User	94
12.8	Update User	96
12.9	Get Own User	99
12.10	Delete User	101
12.11	Delete Identity	102
12.12	List Groups	102
12.13	Get Group	104
12.14	Create Group	107
12.15	Update Group	109
12.16	Delete Group	112
12.17	List Service Accounts	114
12.18	Get Service Account	117
12.19	Delete Service Account	119
12.20	Create Service Account	119
13	Indexes	121
13.1	List Command History Index	121
13.2	List Event Index	122
13.3	List Packet Index	123

13.4	List Parameter Index	125
13.5	List Completeness Index	126
13.6	Stream Packet Index	127
13.7	Stream Parameter Index	128
13.8	Stream Command Index	129
13.9	Stream Event Index	130
13.10	Stream Completeness Index	131
13.11	Rebuild Ccsds Index	132
14	Links	133
14.1	List Links	133
14.2	Get Link	133
14.3	Update Link	134
14.4	Enable Link	135
14.5	Disable Link	135
14.6	Reset Link Counters	136
14.7	Subscribe Links	136
15	Management	138
15.1	Get System Info	138
15.2	Subscribe System Info	140
15.3	List Instance Templates	142
15.4	Get Instance Template	143
15.5	List Instances	143
15.6	Subscribe Instances	147
15.7	Get Instance	151
15.8	Create Instance	155
15.9	Reconfigure Instance	159
15.10	Start Instance	162
15.11	Stop Instance	166
15.12	Restart Instance	170
15.13	List Services	174
15.14	Get Service	174
15.15	Start Service	175
15.16	Stop Service	175
16	Mdb Override	176
16.1	List Mdb Overrides	176
16.2	Get Algorithm Overrides	176
16.3	Update Parameter	177
16.4	Update Algorithm	183
17	Mdb	190
17.1	Get Mission Database	190
17.2	Export Java Mission Database	191
17.3	List Space Systems	191
17.4	Get Space System	192
17.5	List Parameters	193
17.6	Get Parameter	200
17.7	Batch Get Parameters	206
17.8	List Containers	211
17.9	Get Container	218
17.10	List Commands	223
17.11	Get Command	231
17.12	List Algorithms	238
17.13	Get Algorithm	244
18	Packets	251
18.1	List Packet Names	251

18.2	List Packets	251
18.3	Get Packet	252
18.4	Stream Packets	253
18.5	Export Packet	254
18.6	Export Packets	254
18.7	Subscribe Packets	255
18.8	Subscribe Containers	256
19	Parameter Archive	257
19.1	Rebuild Range	257
19.2	Delete Partitions	257
19.3	Get Parameter Samples	258
19.4	Get Parameter Ranges	259
19.5	List Parameter History	262
19.6	Get Archived Parameters Info	265
19.7	Get Archived Parameter Segments	267
19.8	Get Archived Parameter Group	268
20	Processing	270
20.1	List Processor Types	270
20.2	List Processors	270
20.3	Get Processor	273
20.4	Delete Processor	275
20.5	Edit Processor	276
20.6	Create Processor	276
20.7	Get Parameter Value	277
20.8	Set Parameter Value	279
20.9	Batch Get Parameter Values	281
20.10	Batch Set Parameter Values	283
20.11	Subscribe TM Statistics	284
20.12	Subscribe Parameters	285
20.13	Subscribe Processors	288
20.14	Get Algorithm Status	291
20.15	Subscribe Algorithm Status	292
20.16	Get Algorithm Trace	293
20.17	Edit Algorithm Trace	296
21	Queue	297
21.1	List Queues	297
21.2	Get Queue	299
21.3	Update Queue	301
21.4	Enable Queue	303
21.5	Disable Queue	305
21.6	Block Queue	307
21.7	Subscribe Queue Statistics	309
21.8	Subscribe Queue Events	311
21.9	List Queued Commands	313
21.10	Update Queue Entry	315
21.11	Accept Command	315
21.12	Reject Command	316
22	Replication	317
22.1	Get Replication Info	317
22.2	Subscribe Replication Info	317
23	Rocks Db	319
23.1	List Tablespace	319
23.2	Backup Database	319
23.3	List Databases	320

23.4	Compact Database	320
23.5	Describe Rocks Db	320
23.6	Describe Database	321
24	Server	322
24.1	Get Server Info	322
24.2	List Routes	323
24.3	List Topics	323
24.4	List Threads	324
24.5	Get Thread	325
24.6	Dump Threads	325
24.7	Get Http Traffic	325
25	Sessions	327
25.1	List Sessions	327
26	Stream Archive	328
26.1	List Parameter Groups	328
26.2	List Parameter History	328
26.3	Stream Parameter Values	331
26.4	Get Parameter Samples	334
26.5	Export Parameter Values	335
27	Table	337
27.1	Execute Sql	337
27.2	Execute Streaming Sql	338
27.3	List Streams	340
27.4	Subscribe Stream Statistics	340
27.5	Get Stream	341
27.6	Subscribe Stream	342
27.7	List Tables	343
27.8	Get Table	344
27.9	Get Table Data	345
27.10	Read Rows	347
27.11	Write Rows	348
27.12	Rebuild Histogram	349
28	Time Correlation	350
28.1	Get Config	350
28.2	Set Config	350
28.3	Get Status	351
28.4	Set Coefficients	352
28.5	Reset	352
28.6	Add Time Of Flight Intervals	353
28.7	Delete Time Of Flight Intervals	353
29	Time	355
29.1	Get Leap Seconds	355
29.2	Set Time	355
29.3	Subscribe Time	356
30	Timeline	358
30.1	Create Item	358
30.2	Get Item	359
30.3	Update Item	361
30.4	List Items	362
30.5	Delete Item	364
30.6	Delete Timeline Group	365
30.7	List Sources	366

30.8 List Tags	366
30.9 Add Band	367
30.10 Get Band	368
30.11 List Bands	369
30.12 Update Band	370
30.13 Delete Band	371
30.14 Add View	372
30.15 Get View	374
30.16 List Views	375
30.17 Update View	376
30.18 Delete View	378

1. API Overview

Yamcs provides an HTTP API allowing external tools to integrate with Yamcs resources. Most HTTP endpoints send and expect JSON messages.

Hint: If you develop in Python consider using the [Python Client](https://docs.yamcs.org/python-yamcs-client/)¹ which provides an idiomatic mapping for most of the operations documented here.

HTTP Verbs

The supported HTTP verbs are:

GET Retrieve a resource

POST
Create a new resource

PATCH
Update an existing resource

DELETE
Delete a resource

Time

All timestamps are returned as UTC and formatted according to ISO 8601. For example:

2015-08-26T08:08:40.724Z 2015-08-26

Error Handling

When an exception is caught while handling an HTTP request, the server provides feedback to the client by returning a generic exception message:

```
{
  "exception" : {
    "type": string, // Short message
    "msg": string // Long message
  }
}
```

Clients should check on whether the status code is between 200 and 299, and if not, interpret the response with the above structure.

¹ <https://docs.yamcs.org/python-yamcs-client/>

CORS

Cross-origin Resource Sharing (CORS) allows access to the Yamcs HTTP API from a remotely hosted web page. This is the HTML5 way of bypassing the self-origin policy typically enforced by browsers. With CORS, the browser will issue a preflight request to Yamcs to verify that it allows browser requests from the originating web page.

CORS is off by default on Yamcs Server, but available through configuration.

Response Filtering

Responses can be filtered using the query parameter `fields`, or alternatively by setting the HTTP header `X-Yamcs-Fields`. This is usually not needed, because when unspecified all fields are returned. Some clients may anyway want to be explicit, for optimizing the message sizes.

Field names are applicable to the top-level response message, and multiple fields can be separated by commas. Methods that return a list of messages apply the mask to each of the listed resources. Field paths can be of arbitrary depth separated by dots. Only the last part can refer to a repeated field.

Some examples:

Return information on the *simulator* instance, but include only the name and state fields:

```
curl 'localhost:8090/api/instances/simulator?fields=name,state'
```

Return a list of all instances, but include only the name and state fields:

```
curl 'localhost:8090/api/instances?fields=name,state'
```

JSON

All API methods are designed for JSON-over-HTTP. Matching type definitions in this documentation are written in TypeScript syntax because of its high readability. Note however that we do not currently mark parameters as optional (?).

Protobuf

As an alternative to JSON, most endpoints also support Google Protocol Buffers for a lighter footprint. To mark a request as Protobuf, set this HTTP header:

```
Content-Type: application/protobuf
```

If you also want server to respond with Protobuf messages, add the `Accept` header:

```
Accept: application/protobuf
```

The proto files are [available on GitHub](https://github.com/yamcs/yamcs/blob/master/yamcs-api/src/main/proto/yamcs/protobuf)². Using the `protoc` compiler, client code can be generated for Java, Python, C++ and more.

If the response status is not between 200 and 299, deserialize the response as of type `yamcs.api.ExceptionMessage`.

² <https://github.com/yamcs/yamcs/blob/master/yamcs-api/src/main/proto/yamcs/protobuf>

2. WebSocket

Yamcs provides a WebSocket API for data subscriptions. A typical use case would be a display tool subscribing to parameter updates. But you could also subscribe to realtime events, alarms or even raw packets.

WebSocket allows to upgrade a regular HTTP connection to a bi-directional communication channel. Yamcs supports an RPC-style API over this channel where clients choose what topics they want to subscribe (or unsubscribe) by sending a request in a specific format.

2.1 Connection

WebSocket calls should use a URL of the form `http://localhost:8090/api/websocket`

We suggest using a generic library for establishing a WebSocket connection because the protocol is quite involving.

On the server-side, Yamcs supports two WebSocket subprotocols:

1. Textual WebSocket frames encoded in JSON
2. Binary WebSocket frames encoded in Google Protocol Buffers

To select one or the other specify this header on your WebSocket upgrade request:

```
Sec-WebSocket-Protocol: protobuf
```

or:

```
Sec-WebSocket-Protocol: json
```

When unspecified, the server defaults to JSON. These two formats are functionally identical.

Note: For readability purposes, the next sections focus on JSON.

2.2 Client Message

A message sent by the client to Yamcs must always have this general form:

```
{
  "type": string,
  "options": any | undefined,
  "id": number | undefined,
  "call": number | undefined
}
```

Where:

type The message type. Typically this is the topic to subscribe to, but it could also be a built-in like `cancel`.

options

Options specific to the type.

id An optional client-side message identifier. If you specify this, then Yamcs will return it in reply messages. This purpose of this property is to allow clients to correlate replies with the original request. This is necessary because Yamcs does not guarantee in-order delivery of replies with respect to client requests.

We recommend to use an incrementing number. Yamcs does not currently check on continuity, but it is something we may consider later on.

call Where applicable, this must contain the call associated with this message. This should only be used when the client is streaming multiple messages handled by the same call. Client-streaming is rarely used, so chances are that you will never need to use this option.

2.2.1 Built-in Client messages

Cancel

```
{
  "type": "cancel",
  "options": {
    "call": number
  }
}
```

This message allows to cancel an ongoing call. The call to cancel must be specified as part of the message options.

State

```
{
  "type": "state"
}
```

In response to this message, Yamcs will dump a snapshot of the active calls on the current connection. This is intended for debugging reasons.

2.3 Server Messages

A message sent by the Yamcs to the client will always have this general form:

```
{
  "type": string,
  "call": number | undefined,
  "seq": number | undefined,
  "data": any
}
```

Where:

type The message type. Typically this is the topic that was subscribed to, but it could also be a built-in like reply.

call Where applicable, this contains the call identifier for this message. For the typical case of server-streams, all server messages for a single client request, have the same call identifier.

seq This is a sequence counter scoped to the call. The purpose of this is so that client could detect when some messages have been skipped. Yamcs applies a WebSocket-wide mechanism whereby frames

are dropped if the client is not reading fast enough. If enough frames are dropped, the client connection may even be closed.

data Data associated with this type of server message.

2.3.1 Built-in Server messages

Reply

```
{
  "type": "reply",
  "call": number,
  "seq": number,
  "data": {
    "reply_to": number,
    "exception": any | undefined
  }
}
```

This message is sent by the server in response to a topic request. Yamcs guarantees that this reply message is sent before any other topic messages. The field `reply_to` contains a reference to the `id` from the original client message. If there was an error in handling the request, the reply will provide exception details. This is an object that follows the same structure as exceptions on the regular HTTP API.

State

```
{
  "type": "state",
  "data": {
    "calls": [
      {
        "call": number,
        "type": string,
        "options": any | undefined
      },
      ...
    ]
  }
}
```

This message is sent in response to a request of type `state`. It dumps a list of all active calls. The intended use is for debugging issues. Client that support reconnection cannot rely on this information because it will no longer be present when a new connection is established.

2.4 Example

A simple Hello World example would be to subscribe to time updates coming from the server. Assuming that your Yamcs server has an instance called `myproject`, you would send a message like this indicating your interest:

```
{
  "type": "time",
  "id": 1,
  "options": {
    "instance": "myproject"
  }
}
```

To confirm your request, Yamcs will first send you a reply that looks somewhat like this:

```
{
  "type": "reply",
  "call": 3,
  "seq": 72,
  "data": {
    "@type": "/yamcs.api.Reply",
    "reply_to": 1
  }
}
```

As the client, we note that the server has assigned the call identifier 3 to this subscription.

Note: The property `@type` is an artifact generated by Yamcs JSON backend. It specifies the equivalent Protobuf message type of the data object (Yamcs generates JSON based on Protobuf definitions). You should be able to ignore this property because we enforce each message type to be using only a single data message.

Next we receive continued time updates, each in a WebSocket frame:

```
{
  "type": "time",
  "call": 3,
  "seq": 73,
  "data": {
    "@type": "/google.protobuf.Timestamp",
    "value": "2020-05-14T06:44:32.654Z"
  }
}
```

```
{
  "type": "time",
  "call": 3,
  "seq": 74,
  "data": {
    "@type": "/google.protobuf.Timestamp",
    "value": "2020-05-14T06:44:33.656Z"
  }
}
```

Note that each of these updates can be linked to the call identifier 3. If you had multiple subscriptions going on, this would allow you to couple messages to the correct local handler.

Once you're no longer interested to receive updates for this particular call, you can cancel it like this:

```
{
  "type": "cancel",
  "options": {
    "call": 3
  }
}
```

Of course, if you have no plans to use this connection for other calls, you could as well have closed it altogether.

3. Alarms

3.1 List Alarms

List alarms

URI Template

```
GET /api/archive/{instance}/alarms/{name**}
```

{instance}

Yamcs instance name.

{name}**

Filter alarm instances on a specific alarm name (for example: parameter name)

Query Parameters

pos

The zero-based row number at which to start outputting results. Default: 0

limit

The maximum number of returned records per page. Choose this value too high and you risk hitting the maximum response size limit enforced by the server. Default: 100

start

Filter the lower bound of the alarm's trigger time. Specify a date string in ISO 8601 format. This bound is inclusive.

stop

Filter the upper bound of the alarm's trigger time. Specify a date string in ISO 8601 format. This bound is exclusive.

order

The order of the returned results. Can be either asc or desc. The sorting is always by trigger time (i.e. the generation time of the trigger value). Default: desc

Response Type

```
interface ListAlarmsResponse {
    alarms: AlarmData[];
}
```

Related Types

```
// Summary of an alarm applicable for Parameter or Event (possibly
// other in the future) alarms.
// Contains detailed information on the value occurrence that initially
// triggered the alarm, the most severe value since it originally triggered,
// and the latest value at the time of your request.
interface AlarmData {
    type: AlarmType;
    triggerTime: string; // RFC 3339 timestamp

    // For parameter alarms, this is the id of the parameters
    // For event alarms
    // - the id.namespace is /yamcs/event/<EVENT_SOURCE>, unless
    //   EVENT_SOURCE starts with a "/" in which case the namespace
    //   is just the <EVENT_SOURCE>
    // - the id.name is the <EVENT_TYPE>
    id: NamedObjectId;

    // Distinguisher between multiple alarms for the same id
    seqNum: number;
    severity: AlarmSeverity;

    // Number of times the object was in alarm state
    violations: number;

    // Number of samples received for the object
    count: number;
    acknowledgeInfo: AcknowledgeInfo;
    notificationType: AlarmNotificationType;

    // Additional detail in case the alarm is of type PARAMETER
    parameterDetail: ParameterAlarmData;

    // Additional detail in case the alarm is of type EVENT
    eventDetail: EventAlarmData;

    // Whether the alarm will stay triggered even when the process is OK
    latching: boolean;

    // if the process that generated the alarm is ok (i.e. parameter is within limits)
    processOK: boolean;

    // triggered is same with processOK except when the alarm is latching
    triggered: boolean;

    // if the operator has acknowledged the alarm
    acknowledged: boolean;

    // Details in case the alarm was shelved
    shelveInfo: ShelveInfo;
    clearInfo: ClearInfo;

    // Time when the alarm was last updated
    updateTime: string; // RFC 3339 timestamp
}

// Used by external clients to identify an item in the Mission Database
// If namespace is set, then the name is that of an alias, rather than
// the qualified name.
interface NamedObjectId {
    name: string;
    namespace: string;
}
```

(continues on next page)

(continued from previous page)

```
interface AcknowledgeInfo {
    acknowledgedBy: string;
    acknowledgeMessage: string;
    acknowledgeTime: string; // RFC 3339 timestamp
}

interface ParameterAlarmData {
    triggerValue: ParameterValue;
    mostSevereValue: ParameterValue;
    currentValue: ParameterValue;
    parameter: ParameterInfo;
}

interface ParameterValue {
    // Parameter identifier
    id: NamedObjectId;

    // Raw value (uncalibrated)
    rawValue: Value;

    // Engineering value (calibrated)
    engValue: Value;

    // Time of Yams reception
    acquisitionTime: string; // RFC 3339 timestamp

    // Time of generation (~ packet time)
    generationTime: string; // RFC 3339 timestamp
    acquisitionStatus: AcquisitionStatus;

    // Deprecated: this field was originally introduced for compatibility
    // with Airbus CGS/CD-MCS system. It was redundant, because when false,
    // the acquisitionStatus is also set to INVALID.
    processingStatus: boolean;
    monitoringResult: MonitoringResult;
    rangeCondition: RangeCondition;

    // Context-dependent ranges
    alarmRange: AlarmRange[];

    // How long (in milliseconds) this parameter value is valid
    // Note that there is an option when subscribing to parameters to get
    // updated when the parameter values expire.
    expireMillis: string; // String decimal

    // When transferring parameters over WebSocket, this value might be used
    // instead of the id above in order to reduce the bandwidth.
    // Note that the id <-> numericId assignment is only valid in the context
    // of a single WebSocket call.
    numericId: number;
}

// Union type for storing a value
interface Value {
    type: Type;
    floatValue: number;
    doubleValue: number;
    sint32Value: number;
    uint32Value: number;
    binaryValue: string; // Base64
    stringValue: string;
    timestampValue: string; // String decimal
    uint64Value: string; // String decimal
    sint64Value: string; // String decimal
    booleanValue: boolean;
    aggregateValue: AggregateValue;
    arrayValue: Value[];
}

// An aggregate value is an ordered list of (member name, member value).
```

(continues on next page)

```

// Two arrays are used in order to be able to send just the values (since
// the names will not change)
interface AggregateValue {
    name: string[];
    value: Value[];
}

interface AlarmRange {
    level: AlarmLevelType;
    minInclusive: number;
    maxInclusive: number;
    minExclusive: number;
    maxExclusive: number;
}

interface ParameterInfo {
    name: string;
    qualifiedName: string;
    shortDescription: string;
    longDescription: string;
    alias: NamedObjectId[];
    type: ParameterTypeInfo;
    dataSource: DataSourceType;
    usedBy: UsedByInfo;
    ancillaryData: {[key: string]: AncillaryDataInfo};

    // Operations that return aggregate members or array entries
    // may use this field to indicate the path within the parameter.
    path: string[];
}

interface ParameterTypeInfo {
    engType: string;
    dataEncoding: DataEncodingInfo;
    unitSet: UnitInfo[];
    defaultAlarm: AlarmInfo;
    enumValue: EnumValue[];
    absoluteTimeInfo: AbsoluteTimeInfo;
    contextAlarm: ContextAlarmInfo[];
    member: MemberInfo[];
    arrayInfo: ArrayInfo;
    ancillaryData: {[key: string]: AncillaryDataInfo};

    // Provides hints on how to format the engineering
    // value as a string.
    numberFormat: NumberFormatTypeInfo;
}

interface DataEncodingInfo {
    type: Type;
    littleEndian: boolean;
    sizeInBits: number;
    encoding: string;
    defaultCalibrator: CalibratorInfo;
    contextCalibrator: ContextCalibratorInfo[];
}

interface CalibratorInfo {
    polynomialCalibrator: PolynomialCalibratorInfo;
    splineCalibrator: SplineCalibratorInfo;
    javaExpressionCalibrator: JavaExpressionCalibratorInfo;
    type: Type;
}

interface PolynomialCalibratorInfo {
    coefficient: number[];
}

interface SplineCalibratorInfo {
    point: SplinePointInfo[];
}

```

```

interface SplinePointInfo {
    raw: number;
    calibrated: number;
}

interface JavaExpressionCalibratorInfo {
    formula: string;
}

interface ContextCalibratorInfo {
    comparison: ComparisonInfo[];
    calibrator: CalibratorInfo;

    // This can be used in UpdateParameterRequest to pass a context
    // that is parsed on the server, according to the rules in the
    // excel spreadsheet. Either this or a comparison has to be
    // used (not both at the same time)
    context: string;
}

interface ComparisonInfo {
    parameter: ParameterInfo;
    operator: OperatorType;
    value: string;
    argument: ArgumentInfo;
}

interface ArgumentInfo {
    name: string;
    description: string;

    //optional string type = 3;
    initialValue: string;

    // repeated UnitInfo unitSet = 5;
    type: ArgumentTypeInfo;
}

interface ArgumentTypeInfo {
    engType: string;
    dataEncoding: DataEncodingInfo;
    unitSet: UnitInfo[];

    // Enumeration states (only used by enumerated arguments)
    enumValue: EnumValue[];

    // Minimum value (only used by integer and float arguments)
    rangeMin: number;

    // Maximum value (only used by integer and float arguments)
    rangeMax: number;

    // Member information (only used by aggregate arguments)
    member: ArgumentMemberInfo[];

    // String representation of a boolean zero (only used by boolean arguments)
    zeroStringValue: string;

    // String representation of a boolean one (only used by boolean arguments)
    oneStringValue: string;

    // Minimum character count (only used by string arguments)
    minChars: number;

    // Maximum character count (only used by string arguments)
    maxChars: number;

    // True if the engineering type supports signed representation.
    // (only used by integer arguments)
    signed: boolean;
}

```

```

// Minimum byte count (only used by binary arguments)
minBytes: number;

// Maximum character count (only used by binary arguments)
maxBytes: number;
}

interface UnitInfo {
    unit: string;
}

interface EnumValue {
    value: string; // String decimal
    label: string;
    description: string;
}

interface ArgumentMemberInfo {
    name: string;
    shortDescription: string;
    longDescription: string;
    alias: NamedObjectId[];
    type: ArgumentTypeInfo;
}

interface AlarmInfo {
    minViolations: number;
    staticAlarmRange: AlarmRange[];
    enumerationAlarm: EnumerationAlarm[];
}

interface EnumerationAlarm {
    level: AlarmLevelType;
    label: string;
}

interface AbsoluteTimeInfo {
    initialValue: string;
    scale: number;
    offset: number;
    offsetFrom: ParameterInfo;
    epoch: string;
}

interface ContextAlarmInfo {
    comparison: ComparisonInfo[];
    alarm: AlarmInfo;

    // This can be used in UpdateParameterRequest to pass a context
    // that is parsed on the server, according to the rules in the
    // excel spreadsheet. Either this or a comparison has to be
    // used (not both at the same time)
    context: string;
}

interface MemberInfo {
    name: string;
    shortDescription: string;
    longDescription: string;
    alias: NamedObjectId[];
    type: ParameterTypeInfo;
}

interface ArrayInfo {
    type: ParameterTypeInfo;
    dimensions: ParameterDimensionInfo[];
}

interface ParameterDimensionInfo {
    fixedValue: string; // String decimal

```

```

parameter: ParameterInfo;
slope: string; // String decimal
intercept: string; // String decimal
}

interface NumberFormatTypeInfo {
    numberBase: string;
    minimumFractionDigits: number;
    maximumFractionDigits: number;
    minimumIntegerDigits: number;
    maximumIntegerDigits: number;
    negativeSuffix: string;
    positiveSuffix: string;
    negativePrefix: string;
    positivePrefix: string;
    showThousandsGrouping: boolean;
    notation: string;
}

interface UsedByInfo {
    algorithm: AlgorithmInfo[];
    container: ContainerInfo[];
}

interface AlgorithmInfo {
    name: string;
    qualifiedName: string;
    shortDescription: string;
    longDescription: string;
    alias: NamedObjectId[];
    scope: Scope;
    language: string;
    text: string;
    inputParameter: InputParameterInfo[];
    outputParameter: OutputParameterInfo[];
    onParameterUpdate: ParameterInfo[];
    onPeriodicRate: string[]; // String decimal
}

interface InputParameterInfo {
    parameter: ParameterInfo;
    inputName: string;
    parameterInstance: number;
    mandatory: boolean;
    argument: ArgumentInfo;
}

interface OutputParameterInfo {
    parameter: ParameterInfo;
    outputName: string;
}

interface ContainerInfo {
    name: string;
    qualifiedName: string;
    shortDescription: string;
    longDescription: string;
    alias: NamedObjectId[];
    maxInterval: string; // String decimal
    sizeInBits: number;
    baseContainer: ContainerInfo;
    restrictionCriteria: ComparisonInfo[];
    entry: SequenceEntryInfo[];
    usedBy: UsedByInfo;
    ancillaryData: {[key: string]: AncillaryDataInfo};
}

interface SequenceEntryInfo {
    locationInBits: number;
    referenceLocation: ReferenceLocationType;
}

```

```

// For use in sequence containers
container: ContainerInfo;
parameter: ParameterInfo;

// For use in command containers
argument: ArgumentInfo;
fixedValue: FixedValueInfo;
repeat: RepeatInfo;
}

interface FixedValueInfo {
  name: string;
  hexValue: string;
  sizeInBits: number;
}

interface RepeatInfo {
  fixedCount: string; // String decimal
  dynamicCount: ParameterInfo;
  bitsBetween: number;
}

interface EventAlarmData {
  triggerEvent: Event;
  mostSevereEvent: Event;
  currentEvent: Event;
}

interface Event {
  source: string;
  generationTime: string; // RFC 3339 timestamp
  receptionTime: string; // RFC 3339 timestamp
  seqNumber: number;
  type: string;
  message: string;
  severity: EventSeverity;

  // Set by API when event was posted by a user
  createdBy: string;

  // Additional properties
  extra: {[key: string]: string};
}

interface ShelveInfo {
  shelvedBy: string;
  shelveMessage: string;
  shelveTime: string; // RFC 3339 timestamp

  //when the shelving will expire (can be unset which means that it will never expire)
  shelveExpiration: string; // RFC 3339 timestamp
}

interface ClearInfo {
  clearedBy: string;
  clearTime: string; // RFC 3339 timestamp

  //if the alarm has been manually cleared, this is the message provided by the operator
  clearMessage: string;
}

enum AlarmType {
  PARAMETER = "PARAMETER",
  EVENT = "EVENT",
}

enum AlarmSeverity {
  WATCH = "WATCH",
  WARNING = "WARNING",
  DISTRESS = "DISTRESS",
  CRITICAL = "CRITICAL",
}

```

```

    SEVERE = "SEVERE",
}

enum AlarmNotificationType {
    ACTIVE = "ACTIVE",
    TRIGGERED = "TRIGGERED",
    SEVERITY_INCREASED = "SEVERITY_INCREASED",
    VALUE_UPDATED = "VALUE_UPDATED",
    ACKNOWLEDGED = "ACKNOWLEDGED",
    CLEARED = "CLEARED",
    RTN = "RTN",
    SHELVED = "SHELVED",
    UNSHELVED = "UNSHELVED",
    RESET = "RESET",
}

enum Type {
    FLOAT = "FLOAT",
    DOUBLE = "DOUBLE",
    UINT32 = "UINT32",
    SINT32 = "SINT32",
    BINARY = "BINARY",
    STRING = "STRING",
    TIMESTAMP = "TIMESTAMP",
    UINT64 = "UINT64",
    SINT64 = "SINT64",
    BOOLEAN = "BOOLEAN",
    AGGREGATE = "AGGREGATE",
    ARRAY = "ARRAY",

    // Enumerated values have both an integer (sint64Value) and a string representation
    ENUMERATED = "ENUMERATED",
    NONE = "NONE",
}

enum AcquisitionStatus {

    // OK!
    ACQUIRED = "ACQUIRED",

    // No value received so far
    NOT_RECEIVED = "NOT_RECEIVED",

    // Some value has been received but is invalid
    INVALID = "INVALID",

    // The parameter is coming from a packet which has not since updated although it should have been
    EXPIRED = "EXPIRED",
}

enum MonitoringResult {
    DISABLED = "DISABLED",
    IN_LIMITS = "IN_LIMITS",
    WATCH = "WATCH",
    WARNING = "WARNING",
    DISTRESS = "DISTRESS",
    CRITICAL = "CRITICAL",
    SEVERE = "SEVERE",
}

enum RangeCondition {
    LOW = "LOW",
    HIGH = "HIGH",
}

enum AlarmLevelType {
    NORMAL = "NORMAL",
    WATCH = "WATCH",
    WARNING = "WARNING",
    DISTRESS = "DISTRESS",
    CRITICAL = "CRITICAL",
}

```

```

    SEVERE = "SEVERE",
}

enum Type {
    BINARY = "BINARY",
    BOOLEAN = "BOOLEAN",
    FLOAT = "FLOAT",
    INTEGER = "INTEGER",
    STRING = "STRING",
}

enum Type {
    POLYNOMIAL = "POLYNOMIAL",
    SPLINE = "SPLINE",
    MATH_OPERATION = "MATH_OPERATION",
    JAVA_EXPRESSION = "JAVA_EXPRESSION",
}

enum OperatorType {
    EQUAL_TO = "EQUAL_TO",
    NOT_EQUAL_TO = "NOT_EQUAL_TO",
    GREATER_THAN = "GREATER_THAN",
    GREATER_THAN_OR_EQUAL_TO = "GREATER_THAN_OR_EQUAL_TO",
    SMALLER_THAN = "SMALLER_THAN",
    SMALLER_THAN_OR_EQUAL_TO = "SMALLER_THAN_OR_EQUAL_TO",
}

enum AlarmLevelType {
    NORMAL = "NORMAL",
    WATCH = "WATCH",
    WARNING = "WARNING",
    DISTRESS = "DISTRESS",
    CRITICAL = "CRITICAL",
    SEVERE = "SEVERE",
}

enum DataSourceType {
    TELEMETERED = "TELEMETERED",
    DERIVED = "DERIVED",
    CONSTANT = "CONSTANT",
    LOCAL = "LOCAL",
    SYSTEM = "SYSTEM",
    COMMAND = "COMMAND",
    COMMAND_HISTORY = "COMMAND_HISTORY",
    EXTERNAL1 = "EXTERNAL1",
    EXTERNAL2 = "EXTERNAL2",
    EXTERNAL3 = "EXTERNAL3",
}

enum Scope {
    GLOBAL = "GLOBAL",
    COMMAND_VERIFICATION = "COMMAND_VERIFICATION",
    CONTAINER_PROCESSING = "CONTAINER_PROCESSING",
}

enum ReferenceLocationType {
    CONTAINER_START = "CONTAINER_START",
    PREVIOUS_ENTRY = "PREVIOUS_ENTRY",
}

enum EventSeverity {
    INFO = "INFO",
    WARNING = "WARNING",
    ERROR = "ERROR",
    WATCH = "WATCH",
    DISTRESS = "DISTRESS",
    CRITICAL = "CRITICAL",
    SEVERE = "SEVERE",
}

```


3.2 List Processor Alarms

List alarms

URI Template

```
GET /api/processors/{instance}/{processor}/alarms
```

{instance}

{processor}

Response Type

```
interface ListProcessorAlarmsResponse {  
    alarms: AlarmData[];  
}
```

Related Types

```
// Summary of an alarm applicable for Parameter or Event (possibly  
// other in the future) alarms.  
// Contains detailed information on the value occurrence that initially  
// triggered the alarm, the most severe value since it originally triggered,  
// and the latest value at the time of your request.  
interface AlarmData {  
    type: AlarmType;  
    triggerTime: string; // RFC 3339 timestamp  
  
    // For parameter alarms, this is the id of the parameters  
    // For event alarms  
    // - the id.namespace is /yamcs/event/<EVENT_SOURCE>, unless  
    //   EVENT_SOURCE starts with a "/" in which case the namespace  
    //   is just the <EVENT_SOURCE>  
    // - the id.name is the <EVENT_TYPE>  
    id: NamedObjectId;  
  
    // Distinguisher between multiple alarms for the same id  
    seqNum: number;  
    severity: AlarmSeverity;  
  
    // Number of times the object was in alarm state  
    violations: number;  
  
    // Number of samples received for the object  
    count: number;  
    acknowledgeInfo: AcknowledgeInfo;  
    notificationType: AlarmNotificationType;  
  
    // Additional detail in case the alarm is of type PARAMETER  
    parameterDetail: ParameterAlarmData;  
  
    // Additional detail in case the alarm is of type EVENT  
    eventDetail: EventAlarmData;  
  
    // Whether the alarm will stay triggered even when the process is OK  
    latching: boolean;  
  
    // if the process that generated the alarm is ok (i.e. parameter is within limits)  
    processOK: boolean;
```

(continues on next page)

(continued from previous page)

```
// triggered is same with processOK except when the alarm is latching
triggered: boolean;

// if the operator has acknowledged the alarm
acknowledged: boolean;

// Details in case the alarm was shelved
shelveInfo: ShelveInfo;
clearInfo: ClearInfo;

// Time when the alarm was last updated
updateTime: string; // RFC 3339 timestamp
}

// Used by external clients to identify an item in the Mission Database
// If namespace is set, then the name is that of an alias, rather than
// the qualified name.
interface NamedObjectId {
    name: string;
    namespace: string;
}

interface AcknowledgeInfo {
    acknowledgedBy: string;
    acknowledgeMessage: string;
    acknowledgeTime: string; // RFC 3339 timestamp
}

interface ParameterAlarmData {
    triggerValue: ParameterValue;
    mostSevereValue: ParameterValue;
    currentValue: ParameterValue;
    parameter: ParameterInfo;
}

interface ParameterValue {

    // Parameter identifier
    id: NamedObjectId;

    // Raw value (uncalibrated)
    rawValue: Value;

    // Engineering value (calibrated)
    engValue: Value;

    // Time of Yamcs reception
    acquisitionTime: string; // RFC 3339 timestamp

    // Time of generation (~ packet time)
    generationTime: string; // RFC 3339 timestamp
    acquisitionStatus: AcquisitionStatus;

    // Deprecated: this field was originally introduced for compatibility
    // with Airbus CGS/CD-MCS system. It was redundant, because when false,
    // the acquisitionStatus is also set to INVALID.
    processingStatus: boolean;
    monitoringResult: MonitoringResult;
    rangeCondition: RangeCondition;

    // Context-dependent ranges
    alarmRange: AlarmRange[];

    // How long (in milliseconds) this parameter value is valid
    // Note that there is an option when subscribing to parameters to get
    // updated when the parameter values expire.
    expireMillis: string; // String decimal

    // When transferring parameters over WebSocket, this value might be used
    // instead of the id above in order to reduce the bandwidth.
    // Note that the id <-> numericId assignment is only valid in the context
```

(continues on next page)

```

    // of a single WebSocket call.
    numericId: number;
}

// Union type for storing a value
interface Value {
    type: Type;
    floatValue: number;
    doubleValue: number;
    sint32Value: number;
    uint32Value: number;
    binaryValue: string; // Base64
    stringValue: string;
    timestampValue: string; // String decimal
    uint64Value: string; // String decimal
    sint64Value: string; // String decimal
    booleanValue: boolean;
    aggregateValue: AggregateValue;
    arrayValue: Value[];
}

// An aggregate value is an ordered list of (member name, member value).
// Two arrays are used in order to be able to send just the values (since
// the names will not change)
interface AggregateValue {
    name: string[];
    value: Value[];
}

interface AlarmRange {
    level: AlarmLevelType;
    minInclusive: number;
    maxInclusive: number;
    minExclusive: number;
    maxExclusive: number;
}

interface ParameterInfo {
    name: string;
    qualifiedName: string;
    shortDescription: string;
    longDescription: string;
    alias: NamedObjectId[];
    type: ParameterTypeInfo;
    dataSource: DataSourceType;
    usedBy: UsedByInfo;
    ancillaryData: {[key: string]: AncillaryDataInfo};

    // Operations that return aggregate members or array entries
    // may use this field to indicate the path within the parameter.
    path: string[];
}

interface ParameterTypeInfo {
    engType: string;
    dataEncoding: DataEncodingInfo;
    unitSet: UnitInfo[];
    defaultAlarm: AlarmInfo;
    enumValue: EnumValue[];
    absoluteTimeInfo: AbsoluteTimeInfo;
    contextAlarm: ContextAlarmInfo[];
    member: MemberInfo[];
    arrayInfo: ArrayInfo;
    ancillaryData: {[key: string]: AncillaryDataInfo};

    // Provides hints on how to format the engineering
    // value as a string.
    numberFormat: NumberFormatTypeInfo;
}

interface DataEncodingInfo {

```

```

    type: Type;
    littleEndian: boolean;
    sizeInBits: number;
    encoding: string;
    defaultCalibrator: CalibratorInfo;
    contextCalibrator: ContextCalibratorInfo[];
}

interface CalibratorInfo {
    polynomialCalibrator: PolynomialCalibratorInfo;
    splineCalibrator: SplineCalibratorInfo;
    javaExpressionCalibrator: JavaExpressionCalibratorInfo;
    type: Type;
}

interface PolynomialCalibratorInfo {
    coefficient: number[];
}

interface SplineCalibratorInfo {
    point: SplinePointInfo[];
}

interface SplinePointInfo {
    raw: number;
    calibrated: number;
}

interface JavaExpressionCalibratorInfo {
    formula: string;
}

interface ContextCalibratorInfo {
    comparison: ComparisonInfo[];
    calibrator: CalibratorInfo;

    // This can be used in UpdateParameterRequest to pass a context
    // that is parsed on the server, according to the rules in the
    // excel spreadsheet. Either this or a comparison has to be
    // used (not both at the same time)
    context: string;
}

interface ComparisonInfo {
    parameter: ParameterInfo;
    operator: OperatorType;
    value: string;
    argument: ArgumentInfo;
}

interface ArgumentInfo {
    name: string;
    description: string;

    // optional string type = 3;
    initialValue: string;

    // repeated UnitInfo unitSet = 5;
    type: ArgumentTypeInfo;
}

interface ArgumentTypeInfo {
    engType: string;
    dataEncoding: DataEncodingInfo;
    unitSet: UnitInfo[];

    // Enumeration states (only used by enumerated arguments)
    enumValue: EnumValue[];

    // Minimum value (only used by integer and float arguments)
    rangeMin: number;

```

```

// Maximum value (only used by integer and float arguments)
rangeMax: number;

// Member information (only used by aggregate arguments)
member: ArgumentMemberInfo[];

// String representation of a boolean zero (only used by boolean arguments)
zeroStringValue: string;

// String representation of a boolean one (only used by boolean arguments)
oneStringValue: string;

// Minimum character count (only used by string arguments)
minChars: number;

// Maximum character count (only used by string arguments)
maxChars: number;

// True if the engineering type supports signed representation.
// (only used by integer arguments)
signed: boolean;

// Minimum byte count (only used by binary arguments)
minBytes: number;

// Maximum character count (only used by binary arguments)
maxBytes: number;
}

interface UnitInfo {
    unit: string;
}

interface EnumValue {
    value: string; // String decimal
    label: string;
    description: string;
}

interface ArgumentMemberInfo {
    name: string;
    shortDescription: string;
    longDescription: string;
    alias: NamedObjectId[];
    type: ArgumentTypeInfo;
}

interface AlarmInfo {
    minViolations: number;
    staticAlarmRange: AlarmRange[];
    enumerationAlarm: EnumerationAlarm[];
}

interface EnumerationAlarm {
    level: AlarmLevelType;
    label: string;
}

interface AbsoluteTimeInfo {
    initialValue: string;
    scale: number;
    offset: number;
    offsetFrom: ParameterInfo;
    epoch: string;
}

interface ContextAlarmInfo {
    comparison: ComparisonInfo[];
    alarm: AlarmInfo;
}

```

```

// This can be used in UpdateParameterRequest to pass a context
// that is parsed on the server, according to the rules in the
// excel spreadsheet. Either this or a comparison has to be
// used (not both at the same time)
context: string;
}

```

```

interface MemberInfo {
  name: string;
  shortDescription: string;
  longDescription: string;
  alias: NamedObjectId[];
  type: ParameterTypeInfo;
}

```

```

interface ArrayInfo {
  type: ParameterTypeInfo;
  dimensions: ParameterDimensionInfo[];
}

```

```

interface ParameterDimensionInfo {
  fixedValue: string; // String decimal
  parameter: ParameterInfo;
  slope: string; // String decimal
  intercept: string; // String decimal
}

```

```

interface NumberFormatTypeInfo {
  numberBase: string;
  minimumFractionDigits: number;
  maximumFractionDigits: number;
  minimumIntegerDigits: number;
  maximumIntegerDigits: number;
  negativeSuffix: string;
  positiveSuffix: string;
  negativePrefix: string;
  positivePrefix: string;
  showThousandsGrouping: boolean;
  notation: string;
}

```

```

interface UsedByInfo {
  algorithm: AlgorithmInfo[];
  container: ContainerInfo[];
}

```

```

interface AlgorithmInfo {
  name: string;
  qualifiedName: string;
  shortDescription: string;
  longDescription: string;
  alias: NamedObjectId[];
  scope: Scope;
  language: string;
  text: string;
  inputParameter: InputParameterInfo[];
  outputParameter: OutputParameterInfo[];
  onParameterUpdate: ParameterInfo[];
  onPeriodicRate: string[]; // String decimal
}

```

```

interface InputParameterInfo {
  parameter: ParameterInfo;
  inputName: string;
  parameterInstance: number;
  mandatory: boolean;
  argument: ArgumentInfo;
}

```

```

interface OutputParameterInfo {
  parameter: ParameterInfo;
}

```

```

    outputName: string;
}

interface ContainerInfo {
    name: string;
    qualifiedName: string;
    shortDescription: string;
    longDescription: string;
    alias: NamedObjectId[];
    maxInterval: string; // String decimal
    sizeInBits: number;
    baseContainer: ContainerInfo;
    restrictionCriteria: ComparisonInfo[];
    entry: SequenceEntryInfo[];
    usedBy: UsedByInfo;
    ancillaryData: {[key: string]: AncillaryDataInfo};
}

interface SequenceEntryInfo {
    locationInBits: number;
    referenceLocation: ReferenceLocationType;

    // For use in sequence containers
    container: ContainerInfo;
    parameter: ParameterInfo;

    // For use in command containers
    argument: ArgumentInfo;
    fixedValue: FixedValueInfo;
    repeat: RepeatInfo;
}

interface FixedValueInfo {
    name: string;
    hexValue: string;
    sizeInBits: number;
}

interface RepeatInfo {
    fixedCount: string; // String decimal
    dynamicCount: ParameterInfo;
    bitsBetween: number;
}

interface EventAlarmData {
    triggerEvent: Event;
    mostSevereEvent: Event;
    currentEvent: Event;
}

interface Event {
    source: string;
    generationTime: string; // RFC 3339 timestamp
    receptionTime: string; // RFC 3339 timestamp
    seqNumber: number;
    type: string;
    message: string;
    severity: EventSeverity;

    // Set by API when event was posted by a user
    createdBy: string;

    // Additional properties
    extra: {[key: string]: string};
}

interface ShelveInfo {
    shelvedBy: string;
    shelveMessage: string;
    shelveTime: string; // RFC 3339 timestamp

```

(continued from previous page)

```
//when the shelving will expire (can be unset which means that it will never expire)
shelveExpiration: string; // RFC 3339 timestamp
}

interface ClearInfo {
    clearedBy: string;
    clearTime: string; // RFC 3339 timestamp

    //if the alarm has been manually cleared, this is the message provided by the operator
    clearMessage: string;
}

enum AlarmType {
    PARAMETER = "PARAMETER",
    EVENT = "EVENT",
}

enum AlarmSeverity {
    WATCH = "WATCH",
    WARNING = "WARNING",
    DISTRESS = "DISTRESS",
    CRITICAL = "CRITICAL",
    SEVERE = "SEVERE",
}

enum AlarmNotificationType {
    ACTIVE = "ACTIVE",
    TRIGGERED = "TRIGGERED",
    SEVERITY_INCREASED = "SEVERITY_INCREASED",
    VALUE_UPDATED = "VALUE_UPDATED",
    ACKNOWLEDGED = "ACKNOWLEDGED",
    CLEARED = "CLEARED",
    RTN = "RTN",
    SHELVED = "SHELVED",
    UNSHELVED = "UNSHELVED",
    RESET = "RESET",
}

enum Type {
    FLOAT = "FLOAT",
    DOUBLE = "DOUBLE",
    UINT32 = "UINT32",
    SINT32 = "SINT32",
    BINARY = "BINARY",
    STRING = "STRING",
    TIMESTAMP = "TIMESTAMP",
    UINT64 = "UINT64",
    SINT64 = "SINT64",
    BOOLEAN = "BOOLEAN",
    AGGREGATE = "AGGREGATE",
    ARRAY = "ARRAY",

    // Enumerated values have both an integer (sint64Value) and a string representation
    ENUMERATED = "ENUMERATED",
    NONE = "NONE",
}

enum AcquisitionStatus {

    // OK!
    ACQUIRED = "ACQUIRED",

    // No value received so far
    NOT_RECEIVED = "NOT_RECEIVED",

    // Some value has been received but is invalid
    INVALID = "INVALID",

    // The parameter is coming from a packet which has not since updated although it should have been
    EXPIRED = "EXPIRED",
}
```

(continues on next page)


```

enum MonitoringResult {
    DISABLED = "DISABLED",
    IN_LIMITS = "IN_LIMITS",
    WATCH = "WATCH",
    WARNING = "WARNING",
    DISTRESS = "DISTRESS",
    CRITICAL = "CRITICAL",
    SEVERE = "SEVERE",
}

enum RangeCondition {
    LOW = "LOW",
    HIGH = "HIGH",
}

enum AlarmLevelType {
    NORMAL = "NORMAL",
    WATCH = "WATCH",
    WARNING = "WARNING",
    DISTRESS = "DISTRESS",
    CRITICAL = "CRITICAL",
    SEVERE = "SEVERE",
}

enum Type {
    BINARY = "BINARY",
    BOOLEAN = "BOOLEAN",
    FLOAT = "FLOAT",
    INTEGER = "INTEGER",
    STRING = "STRING",
}

enum Type {
    POLYNOMIAL = "POLYNOMIAL",
    SPLINE = "SPLINE",
    MATH_OPERATION = "MATH_OPERATION",
    JAVA_EXPRESSION = "JAVA_EXPRESSION",
}

enum OperatorType {
    EQUAL_TO = "EQUAL_TO",
    NOT_EQUAL_TO = "NOT_EQUAL_TO",
    GREATER_THAN = "GREATER_THAN",
    GREATER_THAN_OR_EQUAL_TO = "GREATER_THAN_OR_EQUAL_TO",
    SMALLER_THAN = "SMALLER_THAN",
    SMALLER_THAN_OR_EQUAL_TO = "SMALLER_THAN_OR_EQUAL_TO",
}

enum AlarmLevelType {
    NORMAL = "NORMAL",
    WATCH = "WATCH",
    WARNING = "WARNING",
    DISTRESS = "DISTRESS",
    CRITICAL = "CRITICAL",
    SEVERE = "SEVERE",
}

enum DataSourceType {
    TELEMETERED = "TELEMETERED",
    DERIVED = "DERIVED",
    CONSTANT = "CONSTANT",
    LOCAL = "LOCAL",
    SYSTEM = "SYSTEM",
    COMMAND = "COMMAND",
    COMMAND_HISTORY = "COMMAND_HISTORY",
    EXTERNAL1 = "EXTERNAL1",
    EXTERNAL2 = "EXTERNAL2",
    EXTERNAL3 = "EXTERNAL3",
}

```

(continued from previous page)

```
enum Scope {
    GLOBAL = "GLOBAL",
    COMMAND_VERIFICATION = "COMMAND_VERIFICATION",
    CONTAINER_PROCESSING = "CONTAINER_PROCESSING",
}

enum ReferenceLocationType {
    CONTAINER_START = "CONTAINER_START",
    PREVIOUS_ENTRY = "PREVIOUS_ENTRY",
}

enum EventSeverity {
    INFO = "INFO",
    WARNING = "WARNING",
    ERROR = "ERROR",
    WATCH = "WATCH",
    DISTRESS = "DISTRESS",
    CRITICAL = "CRITICAL",
    SEVERE = "SEVERE",
}
```

3.3 Edit Alarm

Update an alarm

URI Template

```
PATCH /api/processors/{instance}/{processor}/alarms/{name*}/{seqnum}
```

{instance}
Yamcs instance name.

{processor}
Processor name.

{name*}
Alarm name.

{seqnum}

Request Body

```
interface EditAlarmRequest {

    // **Required.** The state of the alarm.
    // Either ``acknowledged``, ``shelved``, ``unshelved`` or ``cleared``.
    state: string;

    // Message documenting the alarm change.
    comment: string;

    //shelve time in milliseconds (if the state = shelved)
    //can be left out which means it is shelved indefinitely
    shelveDuration: string; // String decimal
}
```

3.4 Acknowledge Alarm

Acknowledge an alarm

URI Template

```
POST /api/processors/{instance}/{processor}/alarms/{alarm*}/{seqnum}:acknowledge
```

{instance}
Yamcs instance name.

{processor}
Processor name.

{alarm*}
Alarm name.

{seqnum}

Request Body

```
interface AcknowledgeAlarmRequest {  
  
    // Message documenting the alarm change.  
    comment: string;  
}
```

3.5 Shelve Alarm

Shelve an alarm

URI Template

```
POST /api/processors/{instance}/{processor}/alarms/{alarm*}/{seqnum}:shelve
```

{instance}
Yamcs instance name.

{processor}
Processor name.

{alarm*}
Alarm name.

{seqnum}

Request Body

```
interface ShelveAlarmRequest {  
  
    // Message documenting the alarm change.  
    comment: string;  
}
```

(continues on next page)

(continued from previous page)

```
//shelve time in milliseconds (if the state = shelved)
//can be left out which means it is shelved indefinitely
shelveDuration: string; // String decimal
}
```

3.6 Unshelve Alarm

Unshelve an alarm

URI Template

```
POST /api/processors/{instance}/{processor}/alarms/{alarm*}/{seqnum}:unshelve
```

{instance}
Yamcs instance name.

{processor}
Processor name.

{alarm*}
Alarm name.

{seqnum}

3.7 Clear Alarm

Clear an alarm

URI Template

```
POST /api/processors/{instance}/{processor}/alarms/{alarm*}/{seqnum}:clear
```

{instance}
Yamcs instance name.

{processor}
Processor name.

{alarm*}
Alarm name.

{seqnum}

Request Body

```
interface ClearAlarmRequest {

    // Message documenting the alarm change.
    comment: string;
}
```

3.8 Subscribe Global Status

Receive alarm status updates

WebSocket

This method requires to upgrade an HTTP connection to WebSocket. See details on [how Yamcs uses WebSocket](#)³.

Use the message type `global-alarm-status`.

Input Type

```
interface SubscribeGlobalStatusRequest {  
  instance: string;  
  processor: string;  
}
```

Output Type

```
interface GlobalAlarmStatus {  
  
  //the number of active unacknoleged alarms  
  unacknowledgedCount: number;  
  
  //true if there is at least one unacknowledged alarm not OK (i.e. latest value of parameter still out of   
↪limits)  
  unacknowledgedActive: boolean;  
  
  //the number of active ackonwledged alarms  
  acknowledgedCount: number;  
  
  //true if there is at least one acknowledged alarm not OK (i.e. latest value of parameter still out of   
↪limits)  
  acknowledgedActive: boolean;  
  
  //the number of shelved alarms  
  shelvedCount: number;  
  
  //true if there is at least one shelved alarm not OK (i.e. latest value of parameter still out of limits)  
  shelvedActive: boolean;  
}
```

3.9 Subscribe Alarms

Receive alarm updates

WebSocket

This method requires to upgrade an HTTP connection to WebSocket. See details on [how Yamcs uses WebSocket](#)⁴.

Use the message type `alarms`.

³ <https://docs.yamcs.org/yamcs-http-api/websocket>

⁴ <https://docs.yamcs.org/yamcs-http-api/websocket>

Input Type

```
interface SubscribeAlarmsRequest {  
  instance: string;  
  processor: string;  
}
```

Output Type

```
// Summary of an alarm applicable for Parameter or Event (possibly  
// other in the future) alarms.  
// Contains detailed information on the value occurrence that initially  
// triggered the alarm, the most severe value since it originally triggered,  
// and the latest value at the time of your request.  
interface AlarmData {  
  type: AlarmType;  
  triggerTime: string; // RFC 3339 timestamp  
  
  // For parameter alarms, this is the id of the parameters  
  // For event alarms  
  // - the id.namespace is /yamcs/event/<EVENT_SOURCE>, unless  
  //   EVENT_SOURCE starts with a "/" in which case the namespace  
  //   is just the <EVENT_SOURCE>  
  // - the id.name is the <EVENT_TYPE>  
  id: NamedObjectId;  
  
  // Distinguisher between multiple alarms for the same id  
  seqNum: number;  
  severity: AlarmSeverity;  
  
  // Number of times the object was in alarm state  
  violations: number;  
  
  // Number of samples received for the object  
  count: number;  
  acknowledgeInfo: AcknowledgeInfo;  
  notificationType: AlarmNotificationType;  
  
  // Additional detail in case the alarm is of type PARAMETER  
  parameterDetail: ParameterAlarmData;  
  
  // Additional detail in case the alarm is of type EVENT  
  eventDetail: EventAlarmData;  
  
  // Whether the alarm will stay triggered even when the process is OK  
  latching: boolean;  
  
  // if the process that generated the alarm is ok (i.e. parameter is within limits)  
  processOK: boolean;  
  
  // triggered is same with processOK except when the alarm is latching  
  triggered: boolean;  
  
  // if the operator has acknowledged the alarm  
  acknowledged: boolean;  
  
  // Details in case the alarm was shelved  
  shelveInfo: ShelveInfo;  
  clearInfo: ClearInfo;  
  
  // Time when the alarm was last updated  
  updateTime: string; // RFC 3339 timestamp  
}
```

Related Types

```
// Used by external clients to identify an item in the Mission Database
// If namespace is set, then the name is that of an alias, rather than
// the qualified name.
interface NamedObjectId {
    name: string;
    namespace: string;
}

interface AcknowledgeInfo {
    acknowledgedBy: string;
    acknowledgeMessage: string;
    acknowledgeTime: string; // RFC 3339 timestamp
}

interface ParameterAlarmData {
    triggerValue: ParameterValue;
    mostSevereValue: ParameterValue;
    currentValue: ParameterValue;
    parameter: ParameterInfo;
}

interface ParameterValue {

    // Parameter identifier
    id: NamedObjectId;

    // Raw value (uncalibrated)
    rawValue: Value;

    // Engineering value (calibrated)
    engValue: Value;

    // Time of Yamcs reception
    acquisitionTime: string; // RFC 3339 timestamp

    // Time of generation (~ packet time)
    generationTime: string; // RFC 3339 timestamp
    acquisitionStatus: AcquisitionStatus;

    // Deprecated: this field was originally introduced for compatibility
    // with Airbus CGS/CD-MCS system. It was redundant, because when false,
    // the acquisitionStatus is also set to INVALID.
    processingStatus: boolean;
    monitoringResult: MonitoringResult;
    rangeCondition: RangeCondition;

    // Context-dependent ranges
    alarmRange: AlarmRange[];

    // How long (in milliseconds) this parameter value is valid
    // Note that there is an option when subscribing to parameters to get
    // updated when the parameter values expire.
    expireMillis: string; // String decimal

    // When transferring parameters over WebSocket, this value might be used
    // instead of the id above in order to reduce the bandwidth.
    // Note that the id <-> numericId assignment is only valid in the context
    // of a single WebSocket call.
    numericId: number;
}

// Union type for storing a value
interface Value {
    type: Type;
    floatValue: number;
    doubleValue: number;
    sint32Value: number;
    uint32Value: number;
    binaryValue: string; // Base64
}
```

(continues on next page)

```

stringValue: string;
timestampValue: string; // String decimal
uint64Value: string; // String decimal
sint64Value: string; // String decimal
booleanValue: boolean;
aggregateValue: AggregateValue;
arrayValue: Value[];
}

// An aggregate value is an ordered list of (member name, member value).
// Two arrays are used in order to be able to send just the values (since
// the names will not change)
interface AggregateValue {
    name: string[];
    value: Value[];
}

interface AlarmRange {
    level: AlarmLevelType;
    minInclusive: number;
    maxInclusive: number;
    minExclusive: number;
    maxExclusive: number;
}

interface ParameterInfo {
    name: string;
    qualifiedName: string;
    shortDescription: string;
    longDescription: string;
    alias: NamedObjectId[];
    type: ParameterTypeInfo;
    dataSource: DataSourceType;
    usedBy: UsedByInfo;
    ancillaryData: {[key: string]: AncillaryDataInfo};

    // Operations that return aggregate members or array entries
    // may use this field to indicate the path within the parameter.
    path: string[];
}

interface ParameterTypeInfo {
    engType: string;
    dataEncoding: DataEncodingInfo;
    unitSet: UnitInfo[];
    defaultAlarm: AlarmInfo;
    enumValue: EnumValue[];
    absoluteTimeInfo: AbsoluteTimeInfo;
    contextAlarm: ContextAlarmInfo[];
    member: MemberInfo[];
    arrayInfo: ArrayInfo;
    ancillaryData: {[key: string]: AncillaryDataInfo};

    // Provides hints on how to format the engineering
    // value as a string.
    numberFormat: NumberFormatTypeInfo;
}

interface DataEncodingInfo {
    type: Type;
    littleEndian: boolean;
    sizeInBits: number;
    encoding: string;
    defaultCalibrator: CalibratorInfo;
    contextCalibrator: ContextCalibratorInfo[];
}

interface CalibratorInfo {
    polynomialCalibrator: PolynomialCalibratorInfo;
    splineCalibrator: SplineCalibratorInfo;
    javaExpressionCalibrator: JavaExpressionCalibratorInfo;
}

```



```

    type: Type;
}

interface PolynomialCalibratorInfo {
    coefficient: number[];
}

interface SplineCalibratorInfo {
    point: SplinePointInfo[];
}

interface SplinePointInfo {
    raw: number;
    calibrated: number;
}

interface JavaExpressionCalibratorInfo {
    formula: string;
}

interface ContextCalibratorInfo {
    comparison: ComparisonInfo[];
    calibrator: CalibratorInfo;

    // This can be used in UpdateParameterRequest to pass a context
    // that is parsed on the server, according to the rules in the
    // excel spreadsheet. Either this or a comparison has to be
    // used (not both at the same time)
    context: string;
}

interface ComparisonInfo {
    parameter: ParameterInfo;
    operator: OperatorType;
    value: string;
    argument: ArgumentInfo;
}

interface ArgumentInfo {
    name: string;
    description: string;

    //optional string type = 3;
    initialValue: string;

    // repeated UnitInfo unitSet = 5;
    type: ArgumentTypeInfo;
}

interface ArgumentTypeInfo {
    engType: string;
    dataEncoding: DataEncodingInfo;
    unitSet: UnitInfo[];

    // Enumeration states (only used by enumerated arguments)
    enumValue: EnumValue[];

    // Minimum value (only used by integer and float arguments)
    rangeMin: number;

    // Maximum value (only used by integer and float arguments)
    rangeMax: number;

    // Member information (only used by aggregate arguments)
    member: ArgumentMemberInfo[];

    // String representation of a boolean zero (only used by boolean arguments)
    zeroStringValue: string;

    // String representation of a boolean one (only used by boolean arguments)
    oneStringValue: string;
}

```

```

// Minimum character count (only used by string arguments)
minChars: number;

// Maximum character count (only used by string arguments)
maxChars: number;

// True if the engineering type supports signed representation.
// (only used by integer arguments)
signed: boolean;

// Minimum byte count (only used by binary arguments)
minBytes: number;

// Maximum character count (only used by binary arguments)
maxBytes: number;
}

interface UnitInfo {
    unit: string;
}

interface EnumValue {
    value: string; // String decimal
    label: string;
    description: string;
}

interface ArgumentMemberInfo {
    name: string;
    shortDescription: string;
    longDescription: string;
    alias: NamedObjectId[];
    type: ArgumentTypeInfo;
}

interface AlarmInfo {
    minViolations: number;
    staticAlarmRange: AlarmRange[];
    enumerationAlarm: EnumerationAlarm[];
}

interface EnumerationAlarm {
    level: AlarmLevelType;
    label: string;
}

interface AbsoluteTimeInfo {
    initialValue: string;
    scale: number;
    offset: number;
    offsetFrom: ParameterInfo;
    epoch: string;
}

interface ContextAlarmInfo {
    comparison: ComparisonInfo[];
    alarm: AlarmInfo;

    // This can be used in UpdateParameterRequest to pass a context
    // that is parsed on the server, according to the rules in the
    // excel spreadsheet. Either this or a comparison has to be
    // used (not both at the same time)
    context: string;
}

interface MemberInfo {
    name: string;
    shortDescription: string;
    longDescription: string;
    alias: NamedObjectId[];

```

```

    type: ParameterTypeInfo;
}

interface ArrayInfo {
    type: ParameterTypeInfo;
    dimensions: ParameterDimensionInfo[];
}

interface ParameterDimensionInfo {
    fixedValue: string; // String decimal
    parameter: ParameterInfo;
    slope: string; // String decimal
    intercept: string; // String decimal
}

interface NumberFormatTypeInfo {
    numberBase: string;
    minimumFractionDigits: number;
    maximumFractionDigits: number;
    minimumIntegerDigits: number;
    maximumIntegerDigits: number;
    negativeSuffix: string;
    positiveSuffix: string;
    negativePrefix: string;
    positivePrefix: string;
    showThousandsGrouping: boolean;
    notation: string;
}

interface UsedByInfo {
    algorithm: AlgorithmInfo[];
    container: ContainerInfo[];
}

interface AlgorithmInfo {
    name: string;
    qualifiedName: string;
    shortDescription: string;
    longDescription: string;
    alias: NamedObjectId[];
    scope: Scope;
    language: string;
    text: string;
    inputParameter: InputParameterInfo[];
    outputParameter: OutputParameterInfo[];
    onParameterUpdate: ParameterInfo[];
    onPeriodicRate: string[]; // String decimal
}

interface InputParameterInfo {
    parameter: ParameterInfo;
    inputName: string;
    parameterInstance: number;
    mandatory: boolean;
    argument: ArgumentInfo;
}

interface OutputParameterInfo {
    parameter: ParameterInfo;
    outputName: string;
}

interface ContainerInfo {
    name: string;
    qualifiedName: string;
    shortDescription: string;
    longDescription: string;
    alias: NamedObjectId[];
    maxInterval: string; // String decimal
    sizeInBits: number;
    baseContainer: ContainerInfo;
}

```

```

restrictionCriteria: ComparisonInfo[];
entry: SequenceEntryInfo[];
usedBy: UsedByInfo;
ancillaryData: {[key: string]: AncillaryDataInfo};
}

interface SequenceEntryInfo {
  locationInBits: number;
  referenceLocation: ReferenceLocationType;

  // For use in sequence containers
  container: ContainerInfo;
  parameter: ParameterInfo;

  // For use in command containers
  argument: ArgumentInfo;
  fixedValue: FixedValueInfo;
  repeat: RepeatInfo;
}

interface FixedValueInfo {
  name: string;
  hexValue: string;
  sizeInBits: number;
}

interface RepeatInfo {
  fixedCount: string; // String decimal
  dynamicCount: ParameterInfo;
  bitsBetween: number;
}

interface EventAlarmData {
  triggerEvent: Event;
  mostSevereEvent: Event;
  currentEvent: Event;
}

interface Event {
  source: string;
  generationTime: string; // RFC 3339 timestamp
  receptionTime: string; // RFC 3339 timestamp
  seqNumber: number;
  type: string;
  message: string;
  severity: EventSeverity;

  // Set by API when event was posted by a user
  createdBy: string;

  // Additional properties
  extra: {[key: string]: string};
}

interface ShelveInfo {
  shelvedBy: string;
  shelveMessage: string;
  shelveTime: string; // RFC 3339 timestamp

  //when the shelving will expire (can be unset which means that it will never expire)
  shelveExpiration: string; // RFC 3339 timestamp
}

interface ClearInfo {
  clearedBy: string;
  clearTime: string; // RFC 3339 timestamp

  //if the alarm has been manually cleared, this is the message provided by the operator
  clearMessage: string;
}

```

```

enum AlarmType {
    PARAMETER = "PARAMETER",
    EVENT = "EVENT",
}

enum AlarmSeverity {
    WATCH = "WATCH",
    WARNING = "WARNING",
    DISTRESS = "DISTRESS",
    CRITICAL = "CRITICAL",
    SEVERE = "SEVERE",
}

enum AlarmNotificationType {
    ACTIVE = "ACTIVE",
    TRIGGERED = "TRIGGERED",
    SEVERITY_INCREASED = "SEVERITY_INCREASED",
    VALUE_UPDATED = "VALUE_UPDATED",
    ACKNOWLEDGED = "ACKNOWLEDGED",
    CLEARED = "CLEARED",
    RTN = "RTN",
    SHELVED = "SHELVED",
    UNSHELVED = "UNSHELVED",
    RESET = "RESET",
}

enum Type {
    FLOAT = "FLOAT",
    DOUBLE = "DOUBLE",
    UINT32 = "UINT32",
    SINT32 = "SINT32",
    BINARY = "BINARY",
    STRING = "STRING",
    TIMESTAMP = "TIMESTAMP",
    UINT64 = "UINT64",
    SINT64 = "SINT64",
    BOOLEAN = "BOOLEAN",
    AGGREGATE = "AGGREGATE",
    ARRAY = "ARRAY",

    // Enumerated values have both an integer (sint64Value) and a string representation
    ENUMERATED = "ENUMERATED",
    NONE = "NONE",
}

enum AcquisitionStatus {

    // OK!
    ACQUIRED = "ACQUIRED",

    // No value received so far
    NOT_RECEIVED = "NOT_RECEIVED",

    // Some value has been received but is invalid
    INVALID = "INVALID",

    // The parameter is coming from a packet which has not since updated although it should have been
    EXPIRED = "EXPIRED",
}

enum MonitoringResult {
    DISABLED = "DISABLED",
    IN_LIMITS = "IN_LIMITS",
    WATCH = "WATCH",
    WARNING = "WARNING",
    DISTRESS = "DISTRESS",
    CRITICAL = "CRITICAL",
    SEVERE = "SEVERE",
}

enum RangeCondition {

```

(continues on next page)

```

    LOW = "LOW",
    HIGH = "HIGH",
}

enum AlarmLevelType {
    NORMAL = "NORMAL",
    WATCH = "WATCH",
    WARNING = "WARNING",
    DISTRESS = "DISTRESS",
    CRITICAL = "CRITICAL",
    SEVERE = "SEVERE",
}

enum Type {
    BINARY = "BINARY",
    BOOLEAN = "BOOLEAN",
    FLOAT = "FLOAT",
    INTEGER = "INTEGER",
    STRING = "STRING",
}

enum Type {
    POLYNOMIAL = "POLYNOMIAL",
    SPLINE = "SPLINE",
    MATH_OPERATION = "MATH_OPERATION",
    JAVA_EXPRESSION = "JAVA_EXPRESSION",
}

enum OperatorType {
    EQUAL_TO = "EQUAL_TO",
    NOT_EQUAL_TO = "NOT_EQUAL_TO",
    GREATER_THAN = "GREATER_THAN",
    GREATER_THAN_OR_EQUAL_TO = "GREATER_THAN_OR_EQUAL_TO",
    SMALLER_THAN = "SMALLER_THAN",
    SMALLER_THAN_OR_EQUAL_TO = "SMALLER_THAN_OR_EQUAL_TO",
}

enum AlarmLevelType {
    NORMAL = "NORMAL",
    WATCH = "WATCH",
    WARNING = "WARNING",
    DISTRESS = "DISTRESS",
    CRITICAL = "CRITICAL",
    SEVERE = "SEVERE",
}

enum DataSourceType {
    TELEMETERED = "TELEMETERED",
    DERIVED = "DERIVED",
    CONSTANT = "CONSTANT",
    LOCAL = "LOCAL",
    SYSTEM = "SYSTEM",
    COMMAND = "COMMAND",
    COMMAND_HISTORY = "COMMAND_HISTORY",
    EXTERNAL1 = "EXTERNAL1",
    EXTERNAL2 = "EXTERNAL2",
    EXTERNAL3 = "EXTERNAL3",
}

enum Scope {
    GLOBAL = "GLOBAL",
    COMMAND_VERIFICATION = "COMMAND_VERIFICATION",
    CONTAINER_PROCESSING = "CONTAINER_PROCESSING",
}

enum ReferenceLocationType {
    CONTAINER_START = "CONTAINER_START",
    PREVIOUS_ENTRY = "PREVIOUS_ENTRY",
}

enum EventSeverity {

```

(continued from previous page)

```
INFO = "INFO",  
WARNING = "WARNING",  
ERROR = "ERROR",  
WATCH = "WATCH",  
DISTRESS = "DISTRESS",  
CRITICAL = "CRITICAL",  
SEVERE = "SEVERE",  
}
```

4. Audit

4.1 List Audit Records

List audit records

URI Template

```
GET /api/audit/records/{instance}
```

{instance}
Yamcs instance.

Query Parameters

limit

The maximum number of returned records per page. Choose this value too high and you risk hitting the maximum response size limit enforced by the server. Default: 100

next

Continuation token returned by a previous page response.

start

Filter the lower bound of the record's time. Specify a date string in ISO 8601 format. This bound is inclusive.

stop

Filter the upper bound of the record's generation time. Specify a date string in ISO 8601 format. This bound is exclusive.

q

Text to search for in the summary.

service

Include only records matching the specified service.

Response Type

```
interface ListAuditRecordsResponse {
  records: AuditRecord[];

  // Token indicating the response is only partial. More results can then
  // be obtained by performing the same request (including all original
  // query parameters) and setting the `next` parameter to this token.
  continuationToken: string;
}
```

Related Types

```
interface AuditRecord {
  time: string; // RFC 3339 timestamp
  service: string;
  method: string;
  user: string;
  summary: string;
  request: {[key: string]: any};
}
```

5. Buckets

Methods related to object storage.

Buckets represent a simple mechanism for storing user objects (binary data chunks such as images, monitoring lists, displays. . .) together with some metadata.

The metadata is represented by simple (key,value) pairs where both key and value are strings.

By default each user has a bucket named `user.username` which can be used without extra privileges. Additional buckets may be created and used if the user has the required privileges. The user bucket will be created automatically when the user tries to access it.

The following limitations are implemented in order to prevent disk over consumption and keep the service responsive:

- The maximum size of an upload including data and metadata is 5MB.
- The maximum number of objects in one bucket is 1000.
- The maximum size of an bucket 100MB (counted as the sum of the size of the objects within the bucket).
- The maximum size of the metadata is 16KB (counted as the sum of the length of the keys and values).

5.1 List Buckets

List buckets

URI Template

```
GET /api/buckets/{instance}
```

{instance}

Yamcs instance name. Or `_global`.

Response Type

```
interface ListBucketsResponse {  
    buckets: BucketInfo[];  
}
```

Related Types

```
interface BucketInfo {  
  
    // Bucket name.  
    name: string;
```

(continues on next page)

(continued from previous page)

```
// Total size in bytes of all objects in the bucket (metadata is not counted)
size: string; // String decimal

// Number of objects in the bucket
numObjects: number;

// Maximum allowed total size of all objects
maxSize: string; // String decimal

// Maximum allowed number of objects
maxObjects: number;

// Creation time of this bucket
created: string; // RFC 3339 timestamp

// Bucket root directory. This field is only set when the
// bucket is mapped to the file system. Therefore it is not
// set for buckets that store objects in RocksDB.
directory: string;
}
```

5.2 Get Bucket

Get a bucket

URI Template

```
GET /api/buckets/{instance}/{bucketName}
```

{instance}

Yamcs instance name. Or `_global`.

{bucketName}

Bucket name.

Response Type

```
interface BucketInfo {

    // Bucket name.
    name: string;

    // Total size in bytes of all objects in the bucket (metadata is not counted)
    size: string; // String decimal

    // Number of objects in the bucket
    numObjects: number;

    // Maximum allowed total size of all objects
    maxSize: string; // String decimal

    // Maximum allowed number of objects
    maxObjects: number;

    // Creation time of this bucket
    created: string; // RFC 3339 timestamp

    // Bucket root directory. This field is only set when the
    // bucket is mapped to the file system. Therefore it is not
    // set for buckets that store objects in RocksDB.
}
```

(continues on next page)

```
directory: string;  
}
```

5.3 Create Bucket

Create a bucket

URI Template

```
POST /api/buckets/{instance}
```

{instance}

Yamcs instance name. Or `_global`.

Request Body

```
interface CreateBucketRequest {  
  
    // Bucket name.  
    name: string;  
}
```

5.4 Delete Bucket

Delete a bucket

Deleting a bucket means also deleting all objects that are part of it.

URI Template

```
DELETE /api/buckets/{instance}/{bucketName}
```

{instance}

Yamcs instance name. Or `_global`.

{bucketName}

Bucket name.

5.5 Get Object

Get an object

The body of the response represents the object data. The `Content-Type` header is set to the content type of the object specified when uploading the object. If no `Content-Type` was specified when creating the object, the `Content-Type` of the response is set to `application/octet-stream`.

URI Template

```
GET /api/buckets/{instance}/{bucketName}/objects/{objectName*}
```

{instance}

Yamcs instance name. Or `_global`.

{bucketName}

Bucket name.

{objectName*}

Object name.

5.6 Upload Object

Upload an object

Simple upload

In case of simple upload, the `objectName` has to be specified as part of the URL and the `Content-Type` header has to be set to the type of the object. The body of the request is the object data.

Form upload

The form based upload can be used to upload an object from an HTML form. In this case the `Content-Type` of the request is set to `multipart/form-data`, and the body will contain at least one part which is the object data. This part includes a filename which is used as the object name as well as a `Content-Type` header. The name attribute for the file part is ignored. Additional parts (which do not specify a filename) will be used as metadata: the name is specified as part of the `Content-Disposition` and the value is the body of the part.

This can be tested with curl using the `-F` option.

Example

```
POST /api/buckets/_global/my_bucket HTTP/1.1
Host: localhost:8090
User-Agent: curl/7.58.0
Accept: */*
Content-Length: 1090
Content-Type: multipart/form-data; boundary=-----7109c709802f7ae4

-----7109c709802f7ae4
Content-Disposition: form-data; name="file"; filename="object/name"
Content-Type: text/plain

[object data]
-----7109c709802f7ae4
Content-Disposition: form-data; name="name1"

value1
-----7109c709802f7ae4
Content-Disposition: form-data; name="name2"

value2
-----7109c709802f7ae4--
```

This will create an object named `object/name` with two metadata properties:

```
{
  "name1": "value1",
  "name2": "value2"
}
```

URI Template

```
POST /api/buckets/{instance}/{bucketName}/objects/{objectName**}
```

{instance}

Yamcs instance name. Or `_global`.

{bucketName}

Bucket name.

If the bucketName is ```user.username``` the bucket will be created automatically if it does not exist. Otherwise the bucket must exist before being used.

{objectName}**

Object name.

Request Body

```
interface HttpBody {
  // The Content-Type header value for this body.
  // If unspecified, defaults to application/octet-stream
  contentType: string;

  // If set, a Content-Disposition header is added
  // to the response. Web agents use this to trigger
  // a download.
  filename: string;

  // The body as raw binary
  data: string; // Base64

  // Any other metadata (used in multipart/form)
  metadata: {[key: string]: string};
}
```

5.7 List Objects

List objects

URI Template

```
GET /api/buckets/{instance}/{bucketName}/objects
```

{instance}

Yamcs instance name. Or `_global`.

{bucketName}

Bucket name.

Query Parameters

delimiter

Return only objects whose name do not contain the delimiter after the prefix. For the other objects, the response contains (in the prefix response parameter) the name truncated after the delimiter. Duplicates are omitted.

Together with `prefix` this parameter provides filtering capabilities. These work similar to Google Cloud Storage and Amazon S3.

The `delimiter` allows the list to work in a directory mode despite the object namespace being flat. For example if the delimiter is set to `/`, then listing the bucket containing objects `"a/b"`, `"a/c"`, `"d"`, `"e"` and `"e/f"` returns objects `"d"` and `"e"` and prefixes `"a/"` and `"e/"`.

prefix

List only objects whose name start with prefix

Response Type

```
interface ListObjectsResponse {  
    prefixes: string[];  
    objects: ObjectInfo[];  
}
```

Related Types

```
interface ObjectInfo {  
  
    // Object name  
    name: string;  
  
    // Creation time  
    created: string; // RFC 3339 timestamp  
  
    // Size in bytes  
    size: string; // String decimal  
    metadata: {[key: string]: string};  
}
```

5.8 Delete Object

Delete an object

URI Template

```
DELETE /api/buckets/{instance}/{bucketName}/objects/{objectName*}
```

{instance}
Yamcs instance name. Or `_global`.

{bucketName}
Bucket name.

{objectName*}
Object name.

6. Clearance

6.1 List Clearances

List clearances

URI Template

```
GET /api/clearances
```

Response Type

```
interface ListClearancesResponse {  
  clearances: ClearanceInfo[];  
}
```

Related Types

```
interface ClearanceInfo {  
  username: string;  
  level: SignificanceLevelType;  
  issuedBy: string;  
  issueTime: string; // RFC 3339 timestamp  
  hasCommandPrivileges: boolean;  
}  
  
enum SignificanceLevelType {  
  NONE = "NONE",  
  WATCH = "WATCH",  
  WARNING = "WARNING",  
  DISTRESS = "DISTRESS",  
  CRITICAL = "CRITICAL",  
  SEVERE = "SEVERE",  
}
```

6.2 Update Clearance

Update a user's clearance

URI Template

```
PATCH /api/clearances/{username}
```



```
{username}
```

Request Body

```
interface UpdateClearanceRequest {  
  level: SignificanceLevelType;  
}
```

Response Type

```
interface ClearanceInfo {  
  username: string;  
  level: SignificanceLevelType;  
  issuedBy: string;  
  issueTime: string; // RFC 3339 timestamp  
  hasCommandPrivileges: boolean;  
}
```

Related Types

```
enum SignificanceLevelType {  
  NONE = "NONE",  
  WATCH = "WATCH",  
  WARNING = "WARNING",  
  DISTRESS = "DISTRESS",  
  CRITICAL = "CRITICAL",  
  SEVERE = "SEVERE",  
}
```

6.3 Delete Clearance

Delete a user's clearance

URI Template

```
DELETE /api/clearances/{username}
```

```
{username}
```

6.4 Subscribe Clearance

Receive updates on own clearance

WebSocket

This method requires to upgrade an HTTP connection to WebSocket. See details on [how Yamcs uses WebSocket⁵](#).

Use the message type `clearance`.

⁵ <https://docs.yamcs.org/yamcs-http-api/websocket>

Output Type

```
interface ClearanceInfo {  
  username: string;  
  level: SignificanceLevelType;  
  issuedBy: string;  
  issueTime: string; // RFC 3339 timestamp  
  hasCommandPrivileges: boolean;  
}
```

Related Types

```
enum SignificanceLevelType {  
  NONE = "NONE",  
  WATCH = "WATCH",  
  WARNING = "WARNING",  
  DISTRESS = "DISTRESS",  
  CRITICAL = "CRITICAL",  
  SEVERE = "SEVERE",  
}
```

7. Commands

7.1 Issue Command

Issue a command

After validating the input parameters, the command is added to the appropriate command queue for further dispatch.

URI Template

```
POST /api/processors/{instance}/{processor}/commands/{name*}
```

{instance}

Yamcs instance name.

{processor}

Processor name.

{name*}

Command name.

Request Body

```
interface IssueCommandRequest {  
  
    // The name/value assignments for this command.  
    args: {[key: string]: any};  
  
    // The name/value assignments for this command.  
    // Deprecated: use ``args`` instead.  
    assignment: Assignment[];  
  
    // The origin of the command. Typically a hostname.  
    origin: string;  
  
    // The sequence number as specified by the origin. This gets  
    // communicated back in command history and command queue entries,  
    // thereby allowing clients to map local with remote command  
    // identities.  
    sequenceNumber: number;  
  
    // Whether a response will be returned without actually issuing  
    // the command. This is useful when debugging commands.  
    // Default ``no``  
    dryRun: boolean;  
  
    // Comment attached to this command.  
    comment: string;  
  
    // Override the stream on which the command should be sent out.  
    // Requires elevated privilege.
```

(continues on next page)

(continued from previous page)

```
stream: string;

// Disable verification of all transmission constrains (if any
// specified in the MDB).
// Requires elevated privilege.
disableTransmissionConstraints: boolean;

// Disable all post transmission verifiers (if any specified in the MDB)
// Requires elevated privilege.
disableVerifiers: boolean;

// Override verifier configuration. Keyed by verifier name
// Requires elevated privilege.
verifierConfig: {[key: string]: VerifierConfig};

// Specify custom options for interpretation by non-core extensions.
// Extensions must register these options against org.yamcs.YamcsServer
extra: {[key: string]: Value};
}
```

Response Type

```
interface IssueCommandResponse {
  id: string;
  generationTime: string; // RFC 3339 timestamp
  origin: string;
  sequenceNumber: number;
  commandName: string;

  // Deprecated. If you require a string representation of this
  // command, you can build it based on the fields ``commandName``
  // and ``assignments``.
  source: string;
  assignments: CommandAssignment[];
  unprocessedBinary: string; // Base64
  binary: string; // Base64
  username: string;
  queue: string;
}
```

Related Types

```
interface Assignment {
  name: string;
  value: string;
}

interface CommandAssignment {
  name: string;
  value: Value;
  userInput: boolean;
}

// Union type for storing a value
interface Value {
  type: Type;
  floatValue: number;
  doubleValue: number;
  sint32Value: number;
  uint32Value: number;
  binaryValue: string; // Base64
  stringValue: string;
  timestampValue: string; // String decimal
  uint64Value: string; // String decimal
  sint64Value: string; // String decimal
}
```

(continues on next page)

(continued from previous page)

```
booleanValue: boolean;
aggregateValue: AggregateValue;
arrayValue: Value[];
}

// An aggregate value is an ordered list of (member name, member value).
// Two arrays are used in order to be able to send just the values (since
// the names will not change)
interface AggregateValue {
    name: string[];
    value: Value[];
}

enum Type {
    FLOAT = "FLOAT",
    DOUBLE = "DOUBLE",
    UINT32 = "UINT32",
    SINT32 = "SINT32",
    BINARY = "BINARY",
    STRING = "STRING",
    TIMESTAMP = "TIMESTAMP",
    UINT64 = "UINT64",
    SINT64 = "SINT64",
    BOOLEAN = "BOOLEAN",
    AGGREGATE = "AGGREGATE",
    ARRAY = "ARRAY",

    // Enumerated values have both an integer (sint64Value) and a string representation
    ENUMERATED = "ENUMERATED",
    NONE = "NONE",
}
```

7.2 Update Command History

Update command history

URI Template

```
POST /api/processors/{instance}/{processor}/commandhistory/{name*}
```

{instance}

Yamcs instance name.

{processor}

Processor name.

{name*}

Command name.

Request Body

```
interface UpdateCommandHistoryRequest {
    id: string;
    attributes: CommandHistoryAttribute[];
}
```

Related Types

```
interface CommandHistoryAttribute {
    name: string;
    value: Value;
    time: string; // String decimal
}

// Union type for storing a value
interface Value {
    type: Type;
    floatValue: number;
    doubleValue: number;
    sint32Value: number;
    uint32Value: number;
    binaryValue: string; // Base64
    stringValue: string;
    timestampValue: string; // String decimal
    uint64Value: string; // String decimal
    sint64Value: string; // String decimal
    booleanValue: boolean;
    aggregateValue: AggregateValue;
    arrayValue: Value[];
}

// An aggregate value is an ordered list of (member name, member value).
// Two arrays are used in order to be able to send just the values (since
// the names will not change)
interface AggregateValue {
    name: string[];
    value: Value[];
}

enum Type {
    FLOAT = "FLOAT",
    DOUBLE = "DOUBLE",
    UINT32 = "UINT32",
    SINT32 = "SINT32",
    BINARY = "BINARY",
    STRING = "STRING",
    TIMESTAMP = "TIMESTAMP",
    UINT64 = "UINT64",
    SINT64 = "SINT64",
    BOOLEAN = "BOOLEAN",
    AGGREGATE = "AGGREGATE",
    ARRAY = "ARRAY",

    // Enumerated values have both an integer (sint64Value) and a string representation
    ENUMERATED = "ENUMERATED",
    NONE = "NONE",
}
```

7.3 List Commands

List commands

URI Template

```
GET /api/archive/{instance}/commands
```

{instance}

Yamcs instance name.

Query Parameters

pos

The zero-based row number at which to start outputting results. Default: 0

This option is deprecated and will be removed in a later version. Use the returned continuationToken instead.

limit

The maximum number of returned records per page. Choose this value too high and you risk hitting the maximum response size limit enforced by the server. Default: 100

order

The order of the returned results. Can be either asc or desc. Default: desc

q

Text to search in the name of the command.

next

Continuation token returned by a previous page response.

start

Filter the lower bound of the command's generation time. Specify a date string in ISO 8601 format. This bound is inclusive.

stop

Filter the upper bound of the command's generation time. Specify a date string in ISO 8601 format. This bound is exclusive.

queue

Filter the results by the used queue.

Response Type

```
interface ListCommandsResponse {  
    entry: CommandHistoryEntry[];  
  
    // Token indicating the response is only partial. More results can then  
    // be obtained by performing the same request (including all original  
    // query parameters) and setting the ``next`` parameter to this token.  
    continuationToken: string;  
}
```

Related Types

```
interface CommandHistoryEntry {  
    id: string;  
    commandName: string;  
    origin: string;  
    sequenceNumber: number;  
    commandId: CommandId;  
}
```

(continues on next page)

```

    attr: CommandHistoryAttribute[];
    generationTime: string; // RFC 3339 timestamp

    // Deprecated. Use the field `assignments` instead.
    assignment: CommandAssignment[];
    assignments: CommandAssignment[];
}

interface CommandId {
    generationTime: string; // String decimal
    origin: string;
    sequenceNumber: number;
    commandName: string;
}

interface CommandHistoryAttribute {
    name: string;
    value: Value;
    time: string; // String decimal
}

// Union type for storing a value
interface Value {
    type: Type;
    floatValue: number;
    doubleValue: number;
    sint32Value: number;
    uint32Value: number;
    binaryValue: string; // Base64
    stringValue: string;
    timestampValue: string; // String decimal
    uint64Value: string; // String decimal
    sint64Value: string; // String decimal
    booleanValue: boolean;
    aggregateValue: AggregateValue;
    arrayValue: Value[];
}

// An aggregate value is an ordered list of (member name, member value).
// Two arrays are used in order to be able to send just the values (since
// the names will not change)
interface AggregateValue {
    name: string[];
    value: Value[];
}

interface CommandAssignment {
    name: string;
    value: Value;
    userInput: boolean;
}

enum Type {
    FLOAT = "FLOAT",
    DOUBLE = "DOUBLE",
    UINT32 = "UINT32",
    SINT32 = "SINT32",
    BINARY = "BINARY",
    STRING = "STRING",
    TIMESTAMP = "TIMESTAMP",
    UINT64 = "UINT64",
    SINT64 = "SINT64",
    BOOLEAN = "BOOLEAN",
    AGGREGATE = "AGGREGATE",
    ARRAY = "ARRAY",

    // Enumerated values have both an integer (sint64Value) and a string representation
    ENUMERATED = "ENUMERATED",
    NONE = "NONE",
}

```


7.4 Get Command

Get a command

URI Template

```
GET /api/archive/{instance}/commands/{id}
```

{instance}

{id}

Response Type

```
interface CommandHistoryEntry {
  id: string;
  commandName: string;
  origin: string;
  sequenceNumber: number;
  commandId: CommandId;
  attr: CommandHistoryAttribute[];
  generationTime: string; // RFC 3339 timestamp

  // Deprecated. Use the field ``assignments`` instead.
  assignment: CommandAssignment[];
  assignments: CommandAssignment[];
}
```

Related Types

```
interface CommandId {
  generationTime: string; // String decimal
  origin: string;
  sequenceNumber: number;
  commandName: string;
}

interface CommandHistoryAttribute {
  name: string;
  value: Value;
  time: string; // String decimal
}

// Union type for storing a value
interface Value {
  type: Type;
  floatValue: number;
  doubleValue: number;
  sint32Value: number;
  uint32Value: number;
  binaryValue: string; // Base64
  stringValue: string;
  timestampValue: string; // String decimal
  uint64Value: string; // String decimal
  sint64Value: string; // String decimal
  booleanValue: boolean;
  aggregateValue: AggregateValue;
  arrayValue: Value[];
}

// An aggregate value is an ordered list of (member name, member value).
```

(continues on next page)

(continued from previous page)

```
// Two arrays are used in order to be able to send just the values (since
// the names will not change)
interface AggregateValue {
    name: string[];
    value: Value[];
}

interface CommandAssignment {
    name: string;
    value: Value;
    userInput: boolean;
}

enum Type {
    FLOAT = "FLOAT",
    DOUBLE = "DOUBLE",
    UINT32 = "UINT32",
    SINT32 = "SINT32",
    BINARY = "BINARY",
    STRING = "STRING",
    TIMESTAMP = "TIMESTAMP",
    UINT64 = "UINT64",
    SINT64 = "SINT64",
    BOOLEAN = "BOOLEAN",
    AGGREGATE = "AGGREGATE",
    ARRAY = "ARRAY",

    // Enumerated values have both an integer (sint64Value) and a string representation
    ENUMERATED = "ENUMERATED",
    NONE = "NONE",
}
```

7.5 Stream Commands

Streams back commands

Warning: This method uses server-streaming. Yamcs sends an unspecified amount of data using chunked transfer encoding.

URI Template

`POST /api/stream-archive/{instance}:streamCommands`

`{instance}`

Request Body

```
interface StreamCommandsRequest {
    start: string; // RFC 3339 timestamp
    stop: string; // RFC 3339 timestamp
    name: string[];
}
```

Response Type

```
interface CommandHistoryEntry {
  id: string;
  commandName: string;
  origin: string;
  sequenceNumber: number;
  commandId: CommandId;
  attr: CommandHistoryAttribute[];
  generationTime: string; // RFC 3339 timestamp

  // Deprecated. Use the field ``assignments`` instead.
  assignment: CommandAssignment[];
  assignments: CommandAssignment[];
}
```

Related Types

```
interface CommandId {
  generationTime: string; // String decimal
  origin: string;
  sequenceNumber: number;
  commandName: string;
}

interface CommandHistoryAttribute {
  name: string;
  value: Value;
  time: string; // String decimal
}

// Union type for storing a value
interface Value {
  type: Type;
  floatValue: number;
  doubleValue: number;
  sint32Value: number;
  uint32Value: number;
  binaryValue: string; // Base64
  stringValue: string;
  timestampValue: string; // String decimal
  uint64Value: string; // String decimal
  sint64Value: string; // String decimal
  booleanValue: boolean;
  aggregateValue: AggregateValue;
  arrayValue: Value[];
}

// An aggregate value is an ordered list of (member name, member value).
// Two arrays are used in order to be able to send just the values (since
// the names will not change)
interface AggregateValue {
  name: string[];
  value: Value[];
}

interface CommandAssignment {
  name: string;
  value: Value;
  userInput: boolean;
}

enum Type {
  FLOAT = "FLOAT",
  DOUBLE = "DOUBLE",
  UINT32 = "UINT32",
  SINT32 = "SINT32",
  BINARY = "BINARY",
  STRING = "STRING",
}
```

(continues on next page)

(continued from previous page)

```
TIMESTAMP = "TIMESTAMP",
UINT64 = "UINT64",
SINT64 = "SINT64",
BOOLEAN = "BOOLEAN",
AGGREGATE = "AGGREGATE",
ARRAY = "ARRAY",

// Enumerated values have both an integer (sint64Value) and a string representation
ENUMERATED = "ENUMERATED",
NONE = "NONE",
}
```

7.6 Subscribe Commands

Receive updates on issued commands

WebSocket

This method requires to upgrade an HTTP connection to WebSocket. See details on [how Yamcs uses WebSocket⁶](#).

Use the message type commands.

Input Type

```
interface SubscribeCommandsRequest {
    instance: string;
    processor: string;
    ignorePastCommands: boolean;
}
```

Output Type

```
interface CommandHistoryEntry {
    id: string;
    commandName: string;
    origin: string;
    sequenceNumber: number;
    commandId: CommandId;
    attr: CommandHistoryAttribute[];
    generationTime: string; // RFC 3339 timestamp

    // Deprecated. Use the field ``assignments`` instead.
    assignment: CommandAssignment[];
    assignments: CommandAssignment[];
}
```

Related Types

```
interface CommandId {
    generationTime: string; // String decimal
    origin: string;
    sequenceNumber: number;
    commandName: string;
}
```

(continues on next page)

⁶ <https://docs.yamcs.org/yamcs-http-api/websocket>

```

}

interface CommandHistoryAttribute {
    name: string;
    value: Value;
    time: string; // String decimal
}

// Union type for storing a value
interface Value {
    type: Type;
    floatValue: number;
    doubleValue: number;
    sint32Value: number;
    uint32Value: number;
    binaryValue: string; // Base64
    stringValue: string;
    timestampValue: string; // String decimal
    uint64Value: string; // String decimal
    sint64Value: string; // String decimal
    booleanValue: boolean;
    aggregateValue: AggregateValue;
    arrayValue: Value[];
}

// An aggregate value is an ordered list of (member name, member value).
// Two arrays are used in order to be able to send just the values (since
// the names will not change)
interface AggregateValue {
    name: string[];
    value: Value[];
}

interface CommandAssignment {
    name: string;
    value: Value;
    userInput: boolean;
}

enum Type {
    FLOAT = "FLOAT",
    DOUBLE = "DOUBLE",
    UINT32 = "UINT32",
    SINT32 = "SINT32",
    BINARY = "BINARY",
    STRING = "STRING",
    TIMESTAMP = "TIMESTAMP",
    UINT64 = "UINT64",
    SINT64 = "SINT64",
    BOOLEAN = "BOOLEAN",
    AGGREGATE = "AGGREGATE",
    ARRAY = "ARRAY",

    // Enumerated values have both an integer (sint64Value) and a string representation
    ENUMERATED = "ENUMERATED",
    NONE = "NONE",
}

```

7.7 Export Command

Export a raw command

URI Template

```
GET /api/archive/{instance}/commands/{id}:export
```

{instance}

{id}

8. Cop1

Methods for virtual channel TC links that have `useCop1: true`. This service contains methods for setting/getting the configuration and performing various operations. In addition, a websocket subscription is available that will allow receiving periodically the status.

8.1 Initialize

Initialize COP-1 in case state is UNINITIALIZED

URI Template

```
POST /api/cop1/{instance}/{link}:initialize
```

{instance}

Yamcs instance name.

{link}

Link name.

Request Body

```
interface InitializeRequest {
  type: InitializationType;

  // Timeout in milliseconds for initialize with CLCW check
  clcwCheckInitializeTimeout: string; // String decimal

  //vR value for initialize with set V(R)
  vR: number;
}
```

Related Types

```
enum InitializationType {

  // CLCW will be expected from the remote system and used to initiate the vS
  WITH_CLCW_CHECK = "WITH_CLCW_CHECK",

  // Initiate without waiting for CLCW
  WITHOUT_CLCW_CHECK = "WITHOUT_CLCW_CHECK",

  // This causes a BC Unlock frame to be sent to the remote system.
  UNLOCK = "UNLOCK",

  // Initiate AD with set V(R). This will cause a BC frame to be sent to the remote system
  SET_VR = "SET_VR",
}
```

8.2 Resume

Resume COP-1 operation in case state is SUSPENDED

URI Template

```
POST /api/cop1/{instance}/{link}:resume
```

{instance}

Yamcs instance name.

{link}

Link name.

8.3 Disable

Disable COP-1 operation

This causes the sent queue to be purged. All TCs from the wait queue, as well as newly received TCs are sent immediately

URI Template

```
POST /api/cop1/{instance}/{link}:disable
```

{instance}

Yamcs instance name.

{link}

Link name.

Request Body

```
interface DisableRequest {  
  
    // If true, all transmitted frames while COP1 is disabled, have the bypass flag set  
    setBypassAll: boolean;  
}
```

8.4 Update Config

Update configuration settings

URI Template

```
PATCH /api/cop1/{instance}/{link}/config
```

{instance}

Yamcs instance name.

{link}

Link name.

Request Body

```
interface Cop1Config {
    vcId: number;

    // If true, the BD frames are sent immediately, without going to the waiting queue
    bdAbsolutePriority: boolean;

    // Maximum size of the sent queue (i.e. how many unacknowledged frames can be in the
    // queue before timing out)
    windowWidth: number;

    // What should happen on timeout: go to SUSPEND or go to UNINITIALIZED
    timeoutType: TimeoutType;

    // How many times the frames are transmitted before timing out
    txLimit: number;

    // How many milliseconds to wait between retransmissions
    t1: string; // String decimal
}
```

Response Type

```
interface Cop1Config {
    vcId: number;

    // If true, the BD frames are sent immediately, without going to the waiting queue
    bdAbsolutePriority: boolean;

    // Maximum size of the sent queue (i.e. how many unacknowledged frames can be in the
    // queue before timing out)
    windowWidth: number;

    // What should happen on timeout: go to SUSPEND or go to UNINITIALIZED
    timeoutType: TimeoutType;

    // How many times the frames are transmitted before timing out
    txLimit: number;

    // How many milliseconds to wait between retransmissions
    t1: string; // String decimal
}
```

Related Types

```
enum TimeoutType {
    UNINITIALIZE = "UNINITIALIZE",
    SUSPEND = "SUSPEND",
}
```

8.5 Get Config

Get COP-1 configuration

URI Template

```
GET /api/cop1/{instance}/{link}/config
```

{instance}
Yamcs instance name.

{link}
Link name.

Response Type

```
interface Cop1Config {
    vcId: number;

    // If true, the BD frames are sent immediately, without going to the waiting queue
    bdAbsolutePriority: boolean;

    // Maximum size of the sent queue (i.e. how many unacknowledged frames can be in the
    // queue before timing out)
    windowWidth: number;

    // What should happen on timeout: go to SUSPEND or go to UNINITIALIZED
    timeoutType: TimeoutType;

    // How many times the frames are transmitted before timing out
    txLimit: number;

    // How many milliseconds to wait between retransmissions
    t1: string; // String decimal
}
```

Related Types

```
enum TimeoutType {
    UNINITIALIZE = "UNINITIALIZE",
    SUSPEND = "SUSPEND",
}
```

8.6 Get Status

Get COP-1 status

URI Template

```
GET /api/cop1/{instance}/{link}/status
```

{instance}
Yamcs instance name.

{link}
Link name.

Response Type

```
interface Cop1Status {

    // Link name for which this status applies.
    // It is present when this message is sent over the websocket as there might
    // be multiple COP-1 links subscribed
    link: string;
}
```

(continues on next page)

(continued from previous page)

```
// If false, all frames are immediately transmitted (i.e. COP-1 is disabled)
cop1Active: boolean;

// Relevant if cop1Active = false -> set the bypass flag on all outgoing frames
setBypassAll: boolean;

// Last received CLCW
clcw: Clcw;

// Current state of FOP-1 state machine, only relevant if cop1Active = true
state: Cop1State;

// V(S) - Transmitter Frame Sequence Number;
vS: number;

// The nR from the previous CLCW
nnR: number;

// Number of TC packets in the wait queue
waitQueueNumTC: number;

// Number of unacknowledged frames in the sent queue
sentQueueNumFrames: number;

// Number of frames in the out queue (waiting to be picked up by the master chain
// multiplexer)
outQueueNumFrames: number;

// How many times the last frame has been transmitted
txCount: number;
}
```

Related Types

```
interface Clcw {
  receptionTime: string; // RFC 3339 timestamp
  lockout: boolean;
  wait: boolean;
  retransmit: boolean;
  nR: number;
}

enum Cop1State {
  ACTIVE = "ACTIVE",
  RETRANSMIT_WITHOUT_WAIT = "RETRANSMIT_WITHOUT_WAIT",
  RETRANSMIT_WITH_WAIT = "RETRANSMIT_WITH_WAIT",
  INITIALIZING_WITHOUT_BC = "INITIALIZING_WITHOUT_BC",
  INITIALIZING_WITH_BC = "INITIALIZING_WITH_BC",
  UNINITIALIZED = "UNINITIALIZED",
  SUSPENDED = "SUSPENDED",
}
```

8.7 Subscribe Status

Receive COP-1 status updates

WebSocket

This method requires to upgrade an HTTP connection to WebSocket. See details on [how Yamcs uses WebSocket⁷](#).

⁷ <https://docs.yamcs.org/yamcs-http-api/websocket>

Use the message type cop1.

Input Type

```
interface SubscribeStatusRequest {  
  
    // Yamcs instance name.  
    instance: string;  
  
    // Link name.  
    link: string;  
}
```

Output Type

```
interface Cop1Status {  
  
    // Link name for which this status applies.  
    // It is present when this message is sent over the websocket as there might  
    // be multiple COP-1 links subscribed  
    link: string;  
  
    // If false, all frames are immediately transmitted (i.e. COP-1 is disabled)  
    cop1Active: boolean;  
  
    // Relevant if cop1Active = false -> set the bypass flag on all outgoing frames  
    setBypassAll: boolean;  
  
    // Last received CLCW  
    clcw: Clcw;  
  
    // Current state of FOP-1 state machine, only relevant if cop1Active = true  
    state: Cop1State;  
  
    // V(S) - Transmitter Frame Sequence Number;  
    vS: number;  
  
    // The nR from the previous CLCW  
    nnR: number;  
  
    // Number of TC packets in the wait queue  
    waitQueueNumTC: number;  
  
    // Number of unacknowledged frames in the sent queue  
    sentQueueNumFrames: number;  
  
    // Number of frames in the out queue (waiting to be picked up by the master chain  
    // multiplexer)  
    outQueueNumFrames: number;  
  
    // How many times the last frame has been transmitted  
    txCount: number;  
}
```

Related Types

```
interface Clcw {  
    receptionTime: string; // RFC 3339 timestamp  
    lockout: boolean;  
    wait: boolean;  
    retransmit: boolean;  
    nR: number;  
}
```

(continues on next page)

(continued from previous page)

```
enum CoplState {  
    ACTIVE = "ACTIVE",  
    RETRANSMIT_WITHOUT_WAIT = "RETRANSMIT_WITHOUT_WAIT",  
    RETRANSMIT_WITH_WAIT = "RETRANSMIT_WITH_WAIT",  
    INITIALIZING_WITHOUT_BC = "INITIALIZING_WITHOUT_BC",  
    INITIALIZING_WITH_BC = "INITIALIZING_WITH_BC",  
    UNINITIALIZED = "UNINITIALIZED",  
    SUSPENDED = "SUSPENDED",  
}
```

9. Database

9.1 List Databases

List databases

URI Template

```
GET /api/databases
```

Response Type

```
interface ListDatabasesResponse {  
  databases: DatabaseInfo[];  
}
```

Related Types

```
interface DatabaseInfo {  
  name: string;  
  path: string;  
  tablespace: string;  
  tables: string[];  
  streams: string[];  
}
```

9.2 Get Database

Get database

URI Template

```
GET /api/databases/{name}
```

{name}

Response Type

```
interface DatabaseInfo {  
    name: string;  
    path: string;  
    tablespace: string;  
    tables: string[];  
    streams: string[];  
}
```

10. Events

10.1 List Events

List events

URI Template

```
GET /api/archive/{instance}/events
```

{instance}

Yamcs instance name.

Query Parameters

pos

The zero-based row number at which to start outputting results. Default: 0

limit

The maximum number of returned records per page. Choose this value too high and you risk hitting the maximum response size limit enforced by the server. Default: 100

order

The order of the returned results. Can be either asc or desc. Default: desc

severity

The minimum severity level of the events. One of info, watch, warning, distress, critical or severe. Default: info

source

The source of the events. Names must match exactly.

next

Continuation token returned by a previous page response.

start

Filter the lower bound of the event's generation time. Specify a date string in ISO 8601 format. This bound is inclusive.

stop

Filter the upper bound of the event's generation time. Specify a date string in ISO 8601 format. This bound is exclusive.

q

Text to search for in the message.

Response Type

```
interface ListEventsResponse {
    event: Event[];

    // Token indicating the response is only partial. More results can then
    // be obtained by performing the same request (including all original
    // query parameters) and setting the ``next`` parameter to this token.
    continuationToken: string;
}
```

Related Types

```
interface Event {
    source: string;
    generationTime: string; // RFC 3339 timestamp
    receptionTime: string; // RFC 3339 timestamp
    seqNumber: number;
    type: string;
    message: string;
    severity: EventSeverity;

    // Set by API when event was posted by a user
    createdBy: string;

    // Additional properties
    extra: {[key: string]: string};
}

enum EventSeverity {
    INFO = "INFO",
    WARNING = "WARNING",
    ERROR = "ERROR",
    WATCH = "WATCH",
    DISTRESS = "DISTRESS",
    CRITICAL = "CRITICAL",
    SEVERE = "SEVERE",
}
```

10.2 Create Event

Create an event

URI Template

```
POST /api/archive/{instance}/events
```

{instance}

Yamcs instance name.

Request Body

```
interface CreateEventRequest {  
  
    // Description of the type of the event. Useful for quick classification or filtering.  
    type: string;  
  
    // **Required.** Event message.  
    message: string;  
  
    // The severity level of the event. One of ``info``, ``watch``, ``warning``,  
    // ``distress``, ``critical`` or ``severe``. Default is ``info``  
    severity: string;  
  
    // Time associated with the event.  
    // If unspecified, this will default to the current mission time.  
    time: string; // RFC 3339 timestamp  
  
    // Source of the event. Useful for grouping events in the archive. Default is  
    // ``User``.  
    source: string;  
  
    // Sequence number of this event. This is primarily used to determine unicity of  
    // events coming from the same source. If not set Yamcs will automatically  
    // assign a sequential number as if every submitted event is unique.  
    sequenceNumber: number;  
  
    // Additional properties  
    extra: {[key: string]: string};  
}
```

Response Type

```
interface Event {  
    source: string;  
    generationTime: string; // RFC 3339 timestamp  
    receptionTime: string; // RFC 3339 timestamp  
    seqNumber: number;  
    type: string;  
    message: string;  
    severity: EventSeverity;  
  
    // Set by API when event was posted by a user  
    createdBy: string;  
  
    // Additional properties  
    extra: {[key: string]: string};  
}
```

Related Types

```
enum EventSeverity {  
    INFO = "INFO",  
    WARNING = "WARNING",  
    ERROR = "ERROR",  
    WATCH = "WATCH",  
    DISTRESS = "DISTRESS",  
    CRITICAL = "CRITICAL",  
    SEVERE = "SEVERE",  
}
```

10.3 List Event Sources

List event sources

URI Template

```
GET /api/archive/{instance}/events/sources
```

{instance}

Yamcs instance name.

Response Type

```
interface ListEventSourcesResponse {  
    source: string[];  
}
```

10.4 Stream Events

Streams back events

Warning: This method uses server-streaming. Yamcs sends an unspecified amount of data using chunked transfer encoding.

URI Template

```
POST /api/stream-archive/{instance}:streamEvents
```

{instance}

Yamcs instance name.

Request Body

```
interface StreamEventsRequest {  
    start: string; // RFC 3339 timestamp  
    stop: string; // RFC 3339 timestamp  
    source: string[];  
    severity: string;  
    q: string;  
}
```

Response Type

```
interface Event {  
    source: string;  
    generationTime: string; // RFC 3339 timestamp  
    receptionTime: string; // RFC 3339 timestamp  
    seqNumber: number;  
    type: string;  
    message: string;  
    severity: EventSeverity;
```

(continues on next page)

(continued from previous page)

```
// Set by API when event was posted by a user
createdBy: string;

// Additional properties
extra: {[key: string]: string};
}
```

Related Types

```
enum EventSeverity {
  INFO = "INFO",
  WARNING = "WARNING",
  ERROR = "ERROR",
  WATCH = "WATCH",
  DISTRESS = "DISTRESS",
  CRITICAL = "CRITICAL",
  SEVERE = "SEVERE",
}
```

10.5 Export Events

Export events in CSV format

Warning: This method uses server-streaming. Yamcs sends an unspecified amount of data using chunked transfer encoding.

URI Template

```
GET /api/archive/{instance}:exportEvents
```

{instance}
Yamcs instance name.

Query Parameters

start

Filter the lower bound of the event's generation time. Specify a date string in ISO 8601 format. This bound is inclusive.

stop

Filter the upper bound of the event's generation time. Specify a date string in ISO 8601 format. This bound is exclusive.

source

The source of the events. Names must match exactly.

severity

The minimum severity level of the events. One of info, watch, warning, distress or severe. Default: info

q

Text to search for in the message.

delimiter

Column delimiter. One of TAB, COMMA or SEMICOLON. Default: TAB.

10.6 Subscribe Events

Receive event updates

WebSocket

This method requires to upgrade an HTTP connection to WebSocket. See details on [how Yamcs uses WebSocket⁸](#).

Use the message type events.

Input Type

```
interface SubscribeEventsRequest {  
    instance: string;  
}
```

Output Type

```
interface Event {  
    source: string;  
    generationTime: string; // RFC 3339 timestamp  
    receptionTime: string; // RFC 3339 timestamp  
    seqNumber: number;  
    type: string;  
    message: string;  
    severity: EventSeverity;  
  
    // Set by API when event was posted by a user  
    createdBy: string;  
  
    // Additional properties  
    extra: {[key: string]: string};  
}
```

Related Types

```
enum EventSeverity {  
    INFO = "INFO",  
    WARNING = "WARNING",  
    ERROR = "ERROR",  
    WATCH = "WATCH",  
    DISTRESS = "DISTRESS",  
    CRITICAL = "CRITICAL",  
}
```

(continues on next page)

⁸ <https://docs.yamcs.org/yamcs-http-api/websocket>

(continued from previous page)

```
SEVERE = "SEVERE",  
}
```

11. File Transfer

11.1 List File Transfer Services

List file transfer services

URI Template

```
GET /api/filetransfer/{instance}/services
```

{instance}

Response Type

```
interface ListFileTransferServicesResponse {  
  services: FileTransferServiceInfo[];  
}
```

Related Types

```
interface FileTransferServiceInfo {  
  instance: string;  
  name: string;  
  localEntities: EntityInfo[];  
  remoteEntities: EntityInfo[];  
  capabilities: FileTransferCapabilities;  
}  
  
interface EntityInfo {  
  name: string;  
  id: string; // String decimal  
}  
  
//This message is used to configure the yacms-web  
interface FileTransferCapabilities {  
  
  //if true, yacms-web shows a button for initiating an upload  
  upload: boolean;  
  
  //if true, yacms-web shows a button for initiating a download  
  download: boolean;  
  
  //if true, yacms-web will allow to chose reliable/non-reliable transfer  
  reliability: boolean;  
  
  //if true, yacms-web will show the section with the destination file/folder name (upload only)  
  remotePath: boolean;  
}
```

11.2 List Transfers

List transfers

URI Template

```
GET /api/filetransfer/{instance}/{serviceName}/transfers
```

{instance}

Yamcs instance name.

{serviceName}

service name

Response Type

```
interface ListTransfersResponse {  
    transfers: TransferInfo[];  
}
```

Related Types

```
//message sent as reponse to the info and also when starting a new transfer  
interface TransferInfo {  
  
    //unique identifier assigned by the file transfer service  
    id: string; // String decimal  
  
    //when the transfer has started. Note that this will not be set for QUEUED transfers.  
    startTime: string; // RFC 3339 timestamp  
    state: TransferState;  
    bucket: string;  
    objectName: string;  
    remotePath: string;  
    direction: TransferDirection;  
    totalSize: string; // String decimal  
    sizeTransferred: string; // String decimal  
  
    //reliable = true -> class 2 transfer  
    //reliable = false -> class 1 transfer  
    reliable: boolean;  
  
    //in case the transaction is failed, this provides more information  
    failureReason: string;  
  
    // valid for CFDP: transaction id;  
    // for the incoming transfers it is assigned by the remote peer so therefore might not be unique  
    transactionId: TransactionId;  
  
    // when the transfer has been created.  
    creationTime: string; // RFC 3339 timestamp  
}  
  
interface TransactionId {  
    sequenceNumber: number;  
    initiatorEntity: string; // String decimal  
}  
  
enum TransferState {  
    RUNNING = "RUNNING",  
    PAUSED = "PAUSED",  
    FAILED = "FAILED",  
    COMPLETED = "COMPLETED",
```

(continues on next page)


```

    QUEUED = "QUEUED",
    CANCELLING = "CANCELLING",
}

enum TransferDirection {
    UPLOAD = "UPLOAD",
    DOWNLOAD = "DOWNLOAD",
}

```

11.3 Get Transfer

Get a transfer

URI Template

```
GET /api/filetransfer/{instance}/{serviceName}/transfers/{id}
```

{instance}
Yamcs instance name.

{serviceName}
service name

{id}
Transfer identifier (assigned by Yamcs)

Response Type

```

//message sent as reponse to the info and also when starting a new transfer
interface TransferInfo {

    //unique identifier assigned by the file transfer service
    id: string; // String decimal

    //when the transfer has started. Note that this will not be set for QUEUED transfers.
    startTime: string; // RFC 3339 timestamp
    state: TransferState;
    bucket: string;
    objectName: string;
    remotePath: string;
    direction: TransferDirection;
    totalSize: string; // String decimal
    sizeTransferred: string; // String decimal

    //reliable = true -> class 2 transfer
    //reliable = false -> class 1 transfer
    reliable: boolean;

    //in case the transaction is failed, this provides more information
    failureReason: string;

    // valid for CFDP: transaction id;
    // for the incoming transfers it is assigned by the remote peer so therefore might not be unique
    transactionId: TransactionId;

    // when the transfer has been created.
    creationTime: string; // RFC 3339 timestamp
}

```

Related Types

```
interface TransactionId {
  sequenceNumber: number;
  initiatorEntity: string; // String decimal
}

enum TransferState {
  RUNNING = "RUNNING",
  PAUSED = "PAUSED",
  FAILED = "FAILED",
  COMPLETED = "COMPLETED",
  QUEUED = "QUEUED",
  CANCELLING = "CANCELLING",
}

enum TransferDirection {
  UPLOAD = "UPLOAD",
  DOWNLOAD = "DOWNLOAD",
}
```

11.4 Create Transfer

Create a transfer

URI Template

```
POST /api/filetransfer/{instance}/{serviceName}/transfers
```

{instance}

{serviceName}
service name

Request Body

```
interface CreateTransferRequest {
  // **Required** One of ``UPLOAD`` or ``DOWNLOAD``.
  direction: TransferDirection;

  // **Required** The bucket containing the local Yamcs object.
  bucket: string;

  // **Required** The object name in Yamcs bucket storage. For UPLOAD transfers,
  // this object must exist and is what Yamcs will transfer to the remote
  // entity. For DOWNLOAD transfers, it refers to the object that
  // Yamcs will write to when downloading from a remote entity.
  objectName: string;

  // **Required** The path at the remote entity. Example: ``a/local/path/some_filename``.
  remotePath: string;
  downloadOptions: DownloadOptions;

  // Configuration options specific to ``UPLOAD`` transfers.
  uploadOptions: UploadOptions;

  //used to derive the source entity id
  source: string;

  //used to derive the destination entity id
}
```

(continues on next page)

```

    destination: string;
}

```

Response Type

```

//message sent as reponse to the info and also when starting a new transfer
interface TransferInfo {

    //unique identifier assigned by the file transfer service
    id: string; // String decimal

    //when the transfer has started. Note that this will not be set for QUEUED transfers.
    startTime: string; // RFC 3339 timestamp
    state: TransferState;
    bucket: string;
    objectName: string;
    remotePath: string;
    direction: TransferDirection;
    totalSize: string; // String decimal
    sizeTransferred: string; // String decimal

    //reliable = true -> class 2 transfer
    //reliable = false -> class 1 transfer
    reliable: boolean;

    //in case the transaction is failed, this provides more information
    failureReason: string;

    // valid for CFDP: transaction id;
    // for the incoming transfers it is assigned by the remote peer so therefore might not be unique
    transactionId: TransactionId;

    // when the transfer has been created.
    creationTime: string; // RFC 3339 timestamp
}

```

Related Types

```

interface DownloadOptions {
}

interface UploadOptions {

    // Set to ``True`` if an already existing destination should be overwritten.
    // Default: ``True``.
    overwrite: boolean;

    // Set to ``True`` if the destination path should be created if it does not exist.
    // Default: ``True``.
    createPath: boolean;

    // Set to ``True`` if reliable (class 2) CFDP transfer should be used,
    // otherwise unreliable (class 1). Default: ``False``.
    reliable: boolean;

    // Introduced in Issue 5 of the CFDP standard for non reliable (class 1) transfers
    // Requests the receiver to send a Finished PDU at the end of the transfer
    closureRequested: boolean;
}

interface TransactionId {
    sequenceNumber: number;
    initiatorEntity: string; // String decimal
}

```

(continued from previous page)

```
enum TransferDirection {
    UPLOAD = "UPLOAD",
    DOWNLOAD = "DOWNLOAD",
}

enum TransferState {
    RUNNING = "RUNNING",
    PAUSED = "PAUSED",
    FAILED = "FAILED",
    COMPLETED = "COMPLETED",
    QUEUED = "QUEUED",
    CANCELLING = "CANCELLING",
}
```

11.5 Pause Transfer

Pause a transfer

URI Template

```
POST /api/filetransfer/{instance}/{serviceName}/transfers/{id}:pause
```

{instance}

Yamcs instance name.

{serviceName}

service name

{id}

Transfer identifier (assigned by Yamcs)

11.6 Cancel Transfer

Cancel a transfer

The ongoing transfer is aborted, partially uploaded/downloaded files are retained.

URI Template

```
POST /api/filetransfer/{instance}/{serviceName}/transfers/{id}:cancel
```

{instance}

Yamcs instance name.

{serviceName}

service name

{id}

Transfer identifier (assigned by Yamcs)

11.7 Resume Transfer

Resume a transfer

URI Template

```
POST /api/filetransfer/{instance}/{serviceName}/transfers/{id}:resume
```

{instance}

Yamcs instance name.

{serviceName}

service name

{id}

Transfer identifier (assigned by Yamcs)

11.8 Subscribe Transfers

Receive transfer updates

WebSocket

This method requires to upgrade an HTTP connection to WebSocket. See details on [how Yamcs uses WebSocket](#)⁹.

Use the message type file-transfers.

Input Type

```
interface SubscribeTransfersRequest {  
  
    // Yamcs instance name.  
    instance: string;  
  
    // service name  
    serviceName: string;  
}
```

Output Type

```
//message sent as reponse to the info and also when starting a new transfer  
interface TransferInfo {  
  
    //unique identifier assigned by the file transfer service  
    id: string; // String decimal  
  
    //when the transfer has started. Note that this will not be set for QUEUED transfers.  
    startTime: string; // RFC 3339 timestamp  
    state: TransferState;  
    bucket: string;  
    objectName: string;  
    remotePath: string;  
    direction: TransferDirection;  
    totalSize: string; // String decimal  
    sizeTransferred: string; // String decimal  
  
    //reliable = true -> class 2 transfer  
    //reliable = false -> class 1 transfer  
    reliable: boolean;
```

(continues on next page)

⁹ <https://docs.yamcs.org/yamcs-http-api/websocket>

(continued from previous page)

```
//in case the transaction is failed, this provides more information
failureReason: string;

// valid for CFDP: transaction id;
// for the incoming transfers it is assigned by the remote peer so therefore might not be unique
transactionId: TransactionId;

// when the transfer has been created.
creationTime: string; // RFC 3339 timestamp
}
```

Related Types

```
interface TransactionId {
    sequenceNumber: number;
    initiatorEntity: string; // String decimal
}

enum TransferState {
    RUNNING = "RUNNING",
    PAUSED = "PAUSED",
    FAILED = "FAILED",
    COMPLETED = "COMPLETED",
    QUEUED = "QUEUED",
    CANCELLING = "CANCELLING",
}

enum TransferDirection {
    UPLOAD = "UPLOAD",
    DOWNLOAD = "DOWNLOAD",
}
```

12. lam

Handles incoming requests related to Identity and Access Management (IAM)

12.1 List Privileges

List privileges

URI Template

```
GET /api/privileges
```

Response Type

```
interface ListPrivilegesResponse {  
    systemPrivileges: string[];  
}
```

12.2 List Roles

List roles

URI Template

```
GET /api/roles
```

Response Type

```
interface ListRolesResponse {  
    roles: RoleInfo[];  
}
```

Related Types

```

interface RoleInfo {

    // Role name
    name: string;

    // Role description
    description: string;

    // System privileges
    systemPrivileges: string[];

    // Object privileges
    objectPrivileges: ObjectPrivilegeInfo[];

    // Whether this role is assigned by default
    default: boolean;
}

interface ObjectPrivilegeInfo {

    // Privilege type
    type: string;
    object: string[];

    // Objects of this type
    objects: string[];
}

```

12.3 Get Role

Get a role

URI Template

```
GET /api/roles/{name}
```

{name}
Role name

Response Type

```

interface RoleInfo {

    // Role name
    name: string;

    // Role description
    description: string;

    // System privileges
    systemPrivileges: string[];

    // Object privileges
    objectPrivileges: ObjectPrivilegeInfo[];

    // Whether this role is assigned by default
    default: boolean;
}

```


Related Types

```
interface ObjectPrivilegeInfo {  
  
    // Privilege type  
    type: string;  
    object: string[];  
  
    // Objects of this type  
    objects: string[];  
}
```

12.4 Delete Role Assignment

Delete a role assignment

URI Template

```
DELETE /api/users/{name}/roles/{role}
```

{name}
Username

{role}
Role name

12.5 List Users

List users

URI Template

```
GET /api/users
```

Response Type

```
interface ListUsersResponse {  
    users: UserInfo[];  
}
```

Related Types

```
interface UserInfo {  
  
    // Username  
    name: string;  
  
    // Displayed name  
    displayName: string;  
  
    // Email address  
    email: string;  
}
```

(continues on next page)

```

// Whether the user may login
active: boolean;

// Whether the user has all privileges
superuser: boolean;

// User that created this user account
createdBy: UserInfo;

// When this user was created
creationTime: string; // RFC 3339 timestamp

// When this user was first activated
confirmationTime: string; // RFC 3339 timestamp

// When this user last logged in
lastLoginTime: string; // RFC 3339 timestamp

// System privileges
systemPrivileges: string[];
systemPrivilege: string[];

// Object privileges
objectPrivileges: ObjectPrivilegeInfo[];
objectPrivilege: ObjectPrivilegeInfo[];

// Groups that this user is member of
groups: GroupInfo[];

// External identities
identities: ExternalIdentityInfo[];

// Assigned roles
roles: RoleInfo[];

// Clearance level. If the command clearance feature is enabled,
// then this user attribute is used as an additional check whether
// the user may send certain commands.
//
// The command clearance feature is disabled by default.
clearance: SignificanceLevelType;
}

interface ObjectPrivilegeInfo {

    // Privilege type
    type: string;
    object: string[];

    // Objects of this type
    objects: string[];
}

interface GroupInfo {

    // Group name
    name: string;

    // Group description
    description: string;

    // Users that are member of this group
    users: UserInfo[];

    // Service accounts that are member of this group
    serviceAccounts: ServiceAccountInfo[];
}

interface ServiceAccountInfo {

```

```

// Service account name
name: string;

// Displayed name
displayName: string;

// Whether the account may login
active: boolean;

// User that created this user account
createdBy: UserInfo;

// When this user was created
creationTime: string; // RFC 3339 timestamp

// When this account was first activated
confirmationTime: string; // RFC 3339 timestamp

// When this account last logged in
lastLoginTime: string; // RFC 3339 timestamp
}

interface ExternalIdentityInfo {

    // External identity
    identity: string;

    // Name of the identity provider
    provider: string;
}

interface RoleInfo {

    // Role name
    name: string;

    // Role description
    description: string;

    // System privileges
    systemPrivileges: string[];

    // Object privileges
    objectPrivileges: ObjectPrivilegeInfo[];

    // Whether this role is assigned by default
    default: boolean;
}

enum SignificanceLevelType {
    NONE = "NONE",
    WATCH = "WATCH",
    WARNING = "WARNING",
    DISTRESS = "DISTRESS",
    CRITICAL = "CRITICAL",
    SEVERE = "SEVERE",
}

```

12.6 Get User

Get a user

URI Template

```
GET /api/users/{name}
```

{name}
Username

Response Type

```
interface UserInfo {  
  
    // Username  
    name: string;  
  
    // Displayed name  
    displayName: string;  
  
    // Email address  
    email: string;  
  
    // Whether the user may login  
    active: boolean;  
  
    // Whether the user has all privileges  
    superuser: boolean;  
  
    // User that created this user account  
    createdBy: UserInfo;  
  
    // When this user was created  
    creationTime: string; // RFC 3339 timestamp  
  
    // When this user was first activated  
    confirmationTime: string; // RFC 3339 timestamp  
  
    // When this user last logged in  
    lastLoginTime: string; // RFC 3339 timestamp  
  
    // System privileges  
    systemPrivileges: string[];  
    systemPrivilege: string[];  
  
    // Object privileges  
    objectPrivileges: ObjectPrivilegeInfo[];  
    objectPrivilege: ObjectPrivilegeInfo[];  
  
    // Groups that this user is member of  
    groups: GroupInfo[];  
  
    // External identities  
    identities: ExternalIdentityInfo[];  
  
    // Assigned roles  
    roles: RoleInfo[];  
  
    // Clearance level. If the command clearance feature is enabled,  
    // then this user attribute is used as an additional check whether  
    // the user may send certain commands.  
    //  
    // The command clearance feature is disabled by default.  
    clearance: SignificanceLevelType;  
}
```

Related Types

```
interface ObjectPrivilegeInfo {  
  
    // Privilege type  
    type: string;  
    object: string[];  
  
    // Objects of this type  
    objects: string[];  
}  
  
interface GroupInfo {  
  
    // Group name  
    name: string;  
  
    // Group description  
    description: string;  
  
    // Users that are member of this group  
    users: UserInfo[];  
  
    // Service accounts that are member of this group  
    serviceAccounts: ServiceAccountInfo[];  
}  
  
interface ServiceAccountInfo {  
  
    // Service account name  
    name: string;  
  
    // Displayed name  
    displayName: string;  
  
    // Whether the account may login  
    active: boolean;  
  
    // User that created this user account  
    createdBy: UserInfo;  
  
    // When this user was created  
    creationTime: string; // RFC 3339 timestamp  
  
    // When this account was first activated  
    confirmationTime: string; // RFC 3339 timestamp  
  
    // When this account last logged in  
    lastLoginTime: string; // RFC 3339 timestamp  
}  
  
interface ExternalIdentityInfo {  
  
    // External identity  
    identity: string;  
  
    // Name of the identity provider  
    provider: string;  
}  
  
interface RoleInfo {  
  
    // Role name  
    name: string;  
  
    // Role description  
    description: string;  
  
    // System privileges  
    systemPrivileges: string[];  
}
```

(continues on next page)

(continued from previous page)

```
// Object privileges
objectPrivileges: ObjectPrivilegeInfo[];

// Whether this role is assigned by default
default: boolean;
}

enum SignificanceLevelType {
    NONE = "NONE",
    WATCH = "WATCH",
    WARNING = "WARNING",
    DISTRESS = "DISTRESS",
    CRITICAL = "CRITICAL",
    SEVERE = "SEVERE",
}
```

12.7 Create User

Create a user

URI Template

POST /api/users

Request Body

```
interface CreateUserRequest {

    // Username
    name: string;

    // Display name
    displayName: string;

    // Email address
    email: string;

    // User password
    password: string;
}
```

Response Type

```
interface UserInfo {

    // Username
    name: string;

    // Displayed name
    displayName: string;

    // Email address
    email: string;

    // Whether the user may login
    active: boolean;

    // Whether the user has all privileges
    superuser: boolean;
}
```

(continues on next page)

```

// User that created this user account
createdBy: UserInfo;

// When this user was created
creationTime: string; // RFC 3339 timestamp

// When this user was first activated
confirmationTime: string; // RFC 3339 timestamp

// When this user last logged in
lastLoginTime: string; // RFC 3339 timestamp

// System privileges
systemPrivileges: string[];
systemPrivilege: string;

// Object privileges
objectPrivileges: ObjectPrivilegeInfo[];
objectPrivilege: ObjectPrivilegeInfo;

// Groups that this user is member of
groups: GroupInfo[];

// External identities
identities: ExternalIdentityInfo[];

// Assigned roles
roles: RoleInfo[];

// Clearance level. If the command clearance feature is enabled,
// then this user attribute is used as an additional check whether
// the user may send certain commands.
//
// The command clearance feature is disabled by default.
clearance: SignificanceLevelType;
}

```

Related Types

```

interface ObjectPrivilegeInfo {

    // Privilege type
    type: string;
    object: string[];

    // Objects of this type
    objects: string[];
}

interface GroupInfo {

    // Group name
    name: string;

    // Group description
    description: string;

    // Users that are member of this group
    users: UserInfo[];

    // Service accounts that are member of this group
    serviceAccounts: ServiceAccountInfo[];
}

interface ServiceAccountInfo {

    // Service account name

```

(continues on next page)

```

name: string;

// Displayed name
displayName: string;

// Whether the account may login
active: boolean;

// User that created this user account
createdBy: UserInfo;

// When this user was created
creationTime: string; // RFC 3339 timestamp

// When this account was first activated
confirmationTime: string; // RFC 3339 timestamp

// When this account last logged in
lastLoginTime: string; // RFC 3339 timestamp
}

interface ExternalIdentityInfo {

    // External identity
    identity: string;

    // Name of the identity provider
    provider: string;
}

interface RoleInfo {

    // Role name
    name: string;

    // Role description
    description: string;

    // System privileges
    systemPrivileges: string[];

    // Object privileges
    objectPrivileges: ObjectPrivilegeInfo[];

    // Whether this role is assigned by default
    default: boolean;
}

enum SignificanceLevelType {
    NONE = "NONE",
    WATCH = "WATCH",
    WARNING = "WARNING",
    DISTRESS = "DISTRESS",
    CRITICAL = "CRITICAL",
    SEVERE = "SEVERE",
}

```

12.8 Update User

Update a user

URI Template

```
PATCH /api/users/{name}
```



```
{name}  
    Username
```

Request Body

```
interface UpdateUserRequest {  
  
    // Display name  
    displayName: string;  
  
    // Email address  
    email: string;  
  
    // Whether the user may login  
    active: boolean;  
  
    // Whether the user has all privileges  
    superuser: boolean;  
  
    // User password  
    password: string;  
  
    // Assigned roles  
    roleAssignment: RoleAssignment;  
}
```

Response Type

```
interface UserInfo {  
  
    // Username  
    name: string;  
  
    // Displayed name  
    displayName: string;  
  
    // Email address  
    email: string;  
  
    // Whether the user may login  
    active: boolean;  
  
    // Whether the user has all privileges  
    superuser: boolean;  
  
    // User that created this user account  
    createdBy: UserInfo;  
  
    // When this user was created  
    creationTime: string; // RFC 3339 timestamp  
  
    // When this user was first activated  
    confirmationTime: string; // RFC 3339 timestamp  
  
    // When this user last logged in  
    lastLoginTime: string; // RFC 3339 timestamp  
  
    // System privileges  
    systemPrivileges: string[];  
    systemPrivilege: string[];  
  
    // Object privileges  
    objectPrivileges: ObjectPrivilegeInfo[];  
    objectPrivilege: ObjectPrivilegeInfo[];  
  
    // Groups that this user is member of  
    groups: GroupInfo[];
```

(continues on next page)

```

// External identities
identities: ExternalIdentityInfo[];

// Assigned roles
roles: RoleInfo[];

// Clearance level. If the command clearance feature is enabled,
// then this user attribute is used as an additional check whether
// the user may send certain commands.
//
// The command clearance feature is disabled by default.
clearance: SignificanceLevelType;
}

```

Related Types

```

interface RoleAssignment {
    roles: string[];
}

interface ObjectPrivilegeInfo {

    // Privilege type
    type: string;
    object: string[];

    // Objects of this type
    objects: string[];
}

interface GroupInfo {

    // Group name
    name: string;

    // Group description
    description: string;

    // Users that are member of this group
    users: UserInfo[];

    // Service accounts that are member of this group
    serviceAccounts: ServiceAccountInfo[];
}

interface ServiceAccountInfo {

    // Service account name
    name: string;

    // Displayed name
    displayName: string;

    // Whether the account may login
    active: boolean;

    // User that created this user account
    createdBy: UserInfo;

    // When this user was created
    creationTime: string; // RFC 3339 timestamp

    // When this account was first activated
    confirmationTime: string; // RFC 3339 timestamp

    // When this account last logged in
    lastLoginTime: string; // RFC 3339 timestamp
}

```

(continues on next page)

(continued from previous page)

```
}

interface ExternalIdentityInfo {

    // External identity
    identity: string;

    // Name of the identity provider
    provider: string;
}

interface RoleInfo {

    // Role name
    name: string;

    // Role description
    description: string;

    // System privileges
    systemPrivileges: string[];

    // Object privileges
    objectPrivileges: ObjectPrivilegeInfo[];

    // Whether this role is assigned by default
    default: boolean;
}

enum SignificanceLevelType {
    NONE = "NONE",
    WATCH = "WATCH",
    WARNING = "WARNING",
    DISTRESS = "DISTRESS",
    CRITICAL = "CRITICAL",
    SEVERE = "SEVERE",
}
```

12.9 Get Own User

Get own user

URI Template

```
GET /api/user
```

Response Type

```
interface UserInfo {

    // Username
    name: string;

    // Displayed name
    displayName: string;

    // Email address
    email: string;

    // Whether the user may login
    active: boolean;
}
```

(continues on next page)

(continued from previous page)

```
// Whether the user has all privileges
superuser: boolean;

// User that created this user account
createdBy: UserInfo;

// When this user was created
creationTime: string; // RFC 3339 timestamp

// When this user was first activated
confirmationTime: string; // RFC 3339 timestamp

// When this user last logged in
lastLoginTime: string; // RFC 3339 timestamp

// System privileges
systemPrivileges: string[];
systemPrivilege: string[];

// Object privileges
objectPrivileges: ObjectPrivilegeInfo[];
objectPrivilege: ObjectPrivilegeInfo[];

// Groups that this user is member of
groups: GroupInfo[];

// External identities
identities: ExternalIdentityInfo[];

// Assigned roles
roles: RoleInfo[];

// Clearance level. If the command clearance feature is enabled,
// then this user attribute is used as an additional check whether
// the user may send certain commands.
//
// The command clearance feature is disabled by default.
clearance: SignificanceLevelType;
}
```

Related Types

```
interface ObjectPrivilegeInfo {

    // Privilege type
    type: string;
    object: string[];

    // Objects of this type
    objects: string[];
}

interface GroupInfo {

    // Group name
    name: string;

    // Group description
    description: string;

    // Users that are member of this group
    users: UserInfo[];

    // Service accounts that are member of this group
    serviceAccounts: ServiceAccountInfo[];
}

interface ServiceAccountInfo {
```

(continues on next page)

```

// Service account name
name: string;

// Displayed name
displayName: string;

// Whether the account may login
active: boolean;

// User that created this user account
createdBy: UserInfo;

// When this user was created
creationTime: string; // RFC 3339 timestamp

// When this account was first activated
confirmationTime: string; // RFC 3339 timestamp

// When this account last logged in
lastLoginTime: string; // RFC 3339 timestamp
}

interface ExternalIdentityInfo {

// External identity
identity: string;

// Name of the identity provider
provider: string;
}

interface RoleInfo {

// Role name
name: string;

// Role description
description: string;

// System privileges
systemPrivileges: string[];

// Object privileges
objectPrivileges: ObjectPrivilegeInfo[];

// Whether this role is assigned by default
default: boolean;
}

enum SignificanceLevelType {
NONE = "NONE",
WATCH = "WATCH",
WARNING = "WARNING",
DISTRESS = "DISTRESS",
CRITICAL = "CRITICAL",
SEVERE = "SEVERE",
}

```

12.10 Delete User

Delete a user

URI Template

```
DELETE /api/users/{name}
```

{name}
Username

12.11 Delete Identity

Delete an external identity

URI Template

```
DELETE /api/users/{name}/identities/{provider}
```

{name}
Username

{provider}
Name of an identity provider

12.12 List Groups

List groups

URI Template

```
GET /api/groups
```

Response Type

```
interface ListGroupsResponse {  
  groups: GroupInfo[];  
}
```

Related Types

```
interface GroupInfo {  
  
  // Group name  
  name: string;  
  
  // Group description  
  description: string;  
  
  // Users that are member of this group  
  users: UserInfo[];  
  
  // Service accounts that are member of this group  
  serviceAccounts: ServiceAccountInfo[];  
}
```

(continues on next page)

```

interface UserInfo {

    // Username
    name: string;

    // Displayed name
    displayName: string;

    // Email address
    email: string;

    // Whether the user may login
    active: boolean;

    // Whether the user has all privileges
    superuser: boolean;

    // User that created this user account
    createdBy: UserInfo;

    // When this user was created
    creationTime: string; // RFC 3339 timestamp

    // When this user was first activated
    confirmationTime: string; // RFC 3339 timestamp

    // When this user last logged in
    lastLoginTime: string; // RFC 3339 timestamp

    // System privileges
    systemPrivileges: string[];
    systemPrivilege: string[];

    // Object privileges
    objectPrivileges: ObjectPrivilegeInfo[];
    objectPrivilege: ObjectPrivilegeInfo[];

    // Groups that this user is member of
    groups: GroupInfo[];

    // External identities
    identities: ExternalIdentityInfo[];

    // Assigned roles
    roles: RoleInfo[];

    // Clearance level. If the command clearance feature is enabled,
    // then this user attribute is used as an additional check whether
    // the user may send certain commands.
    //
    // The command clearance feature is disabled by default.
    clearance: SignificanceLevelType;
}

interface ObjectPrivilegeInfo {

    // Privilege type
    type: string;
    object: string[];

    // Objects of this type
    objects: string[];
}

interface ExternalIdentityInfo {

    // External identity
    identity: string;

    // Name of the identity provider
    provider: string;
}

```

```

}

interface RoleInfo {

    // Role name
    name: string;

    // Role description
    description: string;

    // System privileges
    systemPrivileges: string[];

    // Object privileges
    objectPrivileges: ObjectPrivilegeInfo[];

    // Whether this role is assigned by default
    default: boolean;
}

interface ServiceAccountInfo {

    // Service account name
    name: string;

    // Displayed name
    displayName: string;

    // Whether the account may login
    active: boolean;

    // User that created this user account
    createdBy: UserInfo;

    // When this user was created
    creationTime: string; // RFC 3339 timestamp

    // When this account was first activated
    confirmationTime: string; // RFC 3339 timestamp

    // When this account last logged in
    lastLoginTime: string; // RFC 3339 timestamp
}

enum SignificanceLevelType {
    NONE = "NONE",
    WATCH = "WATCH",
    WARNING = "WARNING",
    DISTRESS = "DISTRESS",
    CRITICAL = "CRITICAL",
    SEVERE = "SEVERE",
}

```

12.13 Get Group

Get a group

URI Template

```
GET /api/groups/{name}
```

{name}
Group name

Response Type

```
interface GroupInfo {  
  
    // Group name  
    name: string;  
  
    // Group description  
    description: string;  
  
    // Users that are member of this group  
    users: UserInfo[];  
  
    // Service accounts that are member of this group  
    serviceAccounts: ServiceAccountInfo[];  
}
```

Related Types

```
interface UserInfo {  
  
    // Username  
    name: string;  
  
    // Displayed name  
    displayName: string;  
  
    // Email address  
    email: string;  
  
    // Whether the user may login  
    active: boolean;  
  
    // Whether the user has all privileges  
    superuser: boolean;  
  
    // User that created this user account  
    createdBy: UserInfo;  
  
    // When this user was created  
    creationTime: string; // RFC 3339 timestamp  
  
    // When this user was first activated  
    confirmationTime: string; // RFC 3339 timestamp  
  
    // When this user last logged in  
    lastLoginTime: string; // RFC 3339 timestamp  
  
    // System privileges  
    systemPrivileges: string[];  
    systemPrivilege: string[];  
  
    // Object privileges  
    objectPrivileges: ObjectPrivilegeInfo[];  
    objectPrivilege: ObjectPrivilegeInfo[];  
  
    // Groups that this user is member of  
    groups: GroupInfo[];  
  
    // External identities  
    identities: ExternalIdentityInfo[];  
  
    // Assigned roles  
    roles: RoleInfo[];  
  
    // Clearance level. If the command clearance feature is enabled,  
    // then this user attribute is used as an additional check whether  
    // the user may send certain commands.  
    //
```

(continues on next page)

```

// The command clearance feature is disabled by default.
clearance: SignificanceLevelType;
}

```

```

interface ObjectPrivilegeInfo {

```

```

    // Privilege type
    type: string;
    object: string[];

    // Objects of this type
    objects: string[];
}

```

```

interface ExternalIdentityInfo {

```

```

    // External identity
    identity: string;

    // Name of the identity provider
    provider: string;
}

```

```

interface RoleInfo {

```

```

    // Role name
    name: string;

    // Role description
    description: string;

    // System privileges
    systemPrivileges: string[];

    // Object privileges
    objectPrivileges: ObjectPrivilegeInfo[];

    // Whether this role is assigned by default
    default: boolean;
}

```

```

interface ServiceAccountInfo {

```

```

    // Service account name
    name: string;

    // Displayed name
    displayName: string;

    // Whether the account may login
    active: boolean;

    // User that created this user account
    createdBy: UserInfo;

    // When this user was created
    creationTime: string; // RFC 3339 timestamp

    // When this account was first activated
    confirmationTime: string; // RFC 3339 timestamp

    // When this account last logged in
    lastLoginTime: string; // RFC 3339 timestamp
}

```

```

enum SignificanceLevelType {

```

```

    NONE = "NONE",
    WATCH = "WATCH",
    WARNING = "WARNING",
    DISTRESS = "DISTRESS",
    CRITICAL = "CRITICAL",
}

```

```
    SEVERE = "SEVERE",
}
```

12.14 Create Group

Create a group

URI Template

POST /api/groups

Request Body

```
interface CreateGroupRequest {

    // Group name
    name: string;

    // Group description
    description: string;

    // Usernames of users that should be added as member
    users: string[];

    // Names of service accounts that should be added as member
    serviceAccounts: string[];
}
```

Response Type

```
interface GroupInfo {

    // Group name
    name: string;

    // Group description
    description: string;

    // Users that are member of this group
    users: UserInfo[];

    // Service accounts that are member of this group
    serviceAccounts: ServiceAccountInfo[];
}
```

Related Types

```
interface UserInfo {

    // Username
    name: string;

    // Displayed name
    displayName: string;

    // Email address
```

(continues on next page)

```

email: string;

// Whether the user may login
active: boolean;

// Whether the user has all privileges
superuser: boolean;

// User that created this user account
createdBy: UserInfo;

// When this user was created
creationTime: string; // RFC 3339 timestamp

// When this user was first activated
confirmationTime: string; // RFC 3339 timestamp

// When this user last logged in
lastLoginTime: string; // RFC 3339 timestamp

// System privileges
systemPrivileges: string[];
systemPrivilege: string[];

// Object privileges
objectPrivileges: ObjectPrivilegeInfo[];
objectPrivilege: ObjectPrivilegeInfo[];

// Groups that this user is member of
groups: GroupInfo[];

// External identities
identities: ExternalIdentityInfo[];

// Assigned roles
roles: RoleInfo[];

// Clearance level. If the command clearance feature is enabled,
// then this user attribute is used as an additional check whether
// the user may send certain commands.
//
// The command clearance feature is disabled by default.
clearance: SignificanceLevelType;
}

interface ObjectPrivilegeInfo {

    // Privilege type
    type: string;
    object: string[];

    // Objects of this type
    objects: string[];
}

interface ExternalIdentityInfo {

    // External identity
    identity: string;

    // Name of the identity provider
    provider: string;
}

interface RoleInfo {

    // Role name
    name: string;

    // Role description
    description: string;
}

```

(continued from previous page)

```
// System privileges
systemPrivileges: string[];

// Object privileges
objectPrivileges: ObjectPrivilegeInfo[];

// Whether this role is assigned by default
default: boolean;
}

interface ServiceAccountInfo {

    // Service account name
    name: string;

    // Displayed name
    displayName: string;

    // Whether the account may login
    active: boolean;

    // User that created this user account
    createdBy: UserInfo;

    // When this user was created
    creationTime: string; // RFC 3339 timestamp

    // When this account was first activated
    confirmationTime: string; // RFC 3339 timestamp

    // When this account last logged in
    lastLoginTime: string; // RFC 3339 timestamp
}

enum SignificanceLevelType {
    NONE = "NONE",
    WATCH = "WATCH",
    WARNING = "WARNING",
    DISTRESS = "DISTRESS",
    CRITICAL = "CRITICAL",
    SEVERE = "SEVERE",
}
```

12.15 Update Group

Update a group

URI Template

```
PATCH /api/groups/{name}
```

{name}
Group name

Request Body

```
interface UpdateGroupRequest {

    // New group name
    newName: string;
}
```

(continues on next page)

```

// Group description
description: string;

// Group members
memberInfo: MemberInfo;
}

```

Response Type

```

interface GroupInfo {

    // Group name
    name: string;

    // Group description
    description: string;

    // Users that are member of this group
    users: UserInfo[];

    // Service accounts that are member of this group
    serviceAccounts: ServiceAccountInfo[];
}

```

Related Types

```

interface MemberInfo {
    users: string[];
    serviceAccounts: string[];
}

interface UserInfo {

    // Username
    name: string;

    // Displayed name
    displayName: string;

    // Email address
    email: string;

    // Whether the user may login
    active: boolean;

    // Whether the user has all privileges
    superuser: boolean;

    // User that created this user account
    createdBy: UserInfo;

    // When this user was created
    creationTime: string; // RFC 3339 timestamp

    // When this user was first activated
    confirmationTime: string; // RFC 3339 timestamp

    // When this user last logged in
    lastLoginTime: string; // RFC 3339 timestamp

    // System privileges
    systemPrivileges: string[];
    systemPrivilege: string[];
}

```

```

// Object privileges
objectPrivileges: ObjectPrivilegeInfo[];
objectPrivilege: ObjectPrivilegeInfo[];

// Groups that this user is member of
groups: GroupInfo[];

// External identities
identities: ExternalIdentityInfo[];

// Assigned roles
roles: RoleInfo[];

// Clearance level. If the command clearance feature is enabled,
// then this user attribute is used as an additional check whether
// the user may send certain commands.
//
// The command clearance feature is disabled by default.
clearance: SignificanceLevelType;
}

interface ObjectPrivilegeInfo {

    // Privilege type
    type: string;
    object: string[];

    // Objects of this type
    objects: string[];
}

interface ExternalIdentityInfo {

    // External identity
    identity: string;

    // Name of the identity provider
    provider: string;
}

interface RoleInfo {

    // Role name
    name: string;

    // Role description
    description: string;

    // System privileges
    systemPrivileges: string[];

    // Object privileges
    objectPrivileges: ObjectPrivilegeInfo[];

    // Whether this role is assigned by default
    default: boolean;
}

interface ServiceAccountInfo {

    // Service account name
    name: string;

    // Displayed name
    displayName: string;

    // Whether the account may login
    active: boolean;

    // User that created this user account
    createdBy: UserInfo;
}

```

(continued from previous page)

```
// When this user was created
creationTime: string; // RFC 3339 timestamp

// When this account was first activated
confirmationTime: string; // RFC 3339 timestamp

// When this account last logged in
lastLoginTime: string; // RFC 3339 timestamp
}

enum SignificanceLevelType {
    NONE = "NONE",
    WATCH = "WATCH",
    WARNING = "WARNING",
    DISTRESS = "DISTRESS",
    CRITICAL = "CRITICAL",
    SEVERE = "SEVERE",
}
```

12.16 Delete Group

Delete a group

URI Template

```
DELETE /api/groups/{name}
```

{name}
Group name

Response Type

```
interface GroupInfo {

    // Group name
    name: string;

    // Group description
    description: string;

    // Users that are member of this group
    users: UserInfo[];

    // Service accounts that are member of this group
    serviceAccounts: ServiceAccountInfo[];
}
```

Related Types

```
interface UserInfo {

    // Username
    name: string;

    // Displayed name
    displayName: string;

    // Email address
```

(continues on next page)


```

email: string;

// Whether the user may login
active: boolean;

// Whether the user has all privileges
superuser: boolean;

// User that created this user account
createdBy: UserInfo;

// When this user was created
creationTime: string; // RFC 3339 timestamp

// When this user was first activated
confirmationTime: string; // RFC 3339 timestamp

// When this user last logged in
lastLoginTime: string; // RFC 3339 timestamp

// System privileges
systemPrivileges: string[];
systemPrivilege: string[];

// Object privileges
objectPrivileges: ObjectPrivilegeInfo[];
objectPrivilege: ObjectPrivilegeInfo[];

// Groups that this user is member of
groups: GroupInfo[];

// External identities
identities: ExternalIdentityInfo[];

// Assigned roles
roles: RoleInfo[];

// Clearance level. If the command clearance feature is enabled,
// then this user attribute is used as an additional check whether
// the user may send certain commands.
//
// The command clearance feature is disabled by default.
clearance: SignificanceLevelType;
}

interface ObjectPrivilegeInfo {

    // Privilege type
    type: string;
    object: string[];

    // Objects of this type
    objects: string[];
}

interface ExternalIdentityInfo {

    // External identity
    identity: string;

    // Name of the identity provider
    provider: string;
}

interface RoleInfo {

    // Role name
    name: string;

    // Role description
    description: string;
}

```

```

// System privileges
systemPrivileges: string[];

// Object privileges
objectPrivileges: ObjectPrivilegeInfo[];

// Whether this role is assigned by default
default: boolean;
}

interface ServiceAccountInfo {

    // Service account name
    name: string;

    // Displayed name
    displayName: string;

    // Whether the account may login
    active: boolean;

    // User that created this user account
    createdBy: UserInfo;

    // When this user was created
    creationTime: string; // RFC 3339 timestamp

    // When this account was first activated
    confirmationTime: string; // RFC 3339 timestamp

    // When this account last logged in
    lastLoginTime: string; // RFC 3339 timestamp
}

enum SignificanceLevelType {
    NONE = "NONE",
    WATCH = "WATCH",
    WARNING = "WARNING",
    DISTRESS = "DISTRESS",
    CRITICAL = "CRITICAL",
    SEVERE = "SEVERE",
}

```

12.17 List Service Accounts

List service accounts

URI Template

```
GET /api/service-accounts
```

Response Type

```

interface ListServiceAccountsResponse {
    serviceAccounts: ServiceAccountInfo[];
}

```

Related Types

```
interface ServiceAccountInfo {  
  
    // Service account name  
    name: string;  
  
    // Displayed name  
    displayName: string;  
  
    // Whether the account may login  
    active: boolean;  
  
    // User that created this user account  
    createdBy: UserInfo;  
  
    // When this user was created  
    creationTime: string; // RFC 3339 timestamp  
  
    // When this account was first activated  
    confirmationTime: string; // RFC 3339 timestamp  
  
    // When this account last logged in  
    lastLoginTime: string; // RFC 3339 timestamp  
}  
  
interface UserInfo {  
  
    // Username  
    name: string;  
  
    // Displayed name  
    displayName: string;  
  
    // Email address  
    email: string;  
  
    // Whether the user may login  
    active: boolean;  
  
    // Whether the user has all privileges  
    superuser: boolean;  
  
    // User that created this user account  
    createdBy: UserInfo;  
  
    // When this user was created  
    creationTime: string; // RFC 3339 timestamp  
  
    // When this user was first activated  
    confirmationTime: string; // RFC 3339 timestamp  
  
    // When this user last logged in  
    lastLoginTime: string; // RFC 3339 timestamp  
  
    // System privileges  
    systemPrivileges: string[];  
    systemPrivilege: string[];  
  
    // Object privileges  
    objectPrivileges: ObjectPrivilegeInfo[];  
    objectPrivilege: ObjectPrivilegeInfo[];  
  
    // Groups that this user is member of  
    groups: GroupInfo[];  
  
    // External identities  
    identities: ExternalIdentityInfo[];  
  
    // Assigned roles  
    roles: RoleInfo[];
```

(continues on next page)

```

// Clearance level. If the command clearance feature is enabled,
// then this user attribute is used as an additional check whether
// the user may send certain commands.
//
// The command clearance feature is disabled by default.
clearance: SignificanceLevelType;
}

interface ObjectPrivilegeInfo {

    // Privilege type
    type: string;
    object: string[];

    // Objects of this type
    objects: string[];
}

interface GroupInfo {

    // Group name
    name: string;

    // Group description
    description: string;

    // Users that are member of this group
    users: UserInfo[];

    // Service accounts that are member of this group
    serviceAccounts: ServiceAccountInfo[];
}

interface ExternalIdentityInfo {

    // External identity
    identity: string;

    // Name of the identity provider
    provider: string;
}

interface RoleInfo {

    // Role name
    name: string;

    // Role description
    description: string;

    // System privileges
    systemPrivileges: string[];

    // Object privileges
    objectPrivileges: ObjectPrivilegeInfo[];

    // Whether this role is assigned by default
    default: boolean;
}

enum SignificanceLevelType {
    NONE = "NONE",
    WATCH = "WATCH",
    WARNING = "WARNING",
    DISTRESS = "DISTRESS",
    CRITICAL = "CRITICAL",
    SEVERE = "SEVERE",
}

```

12.18 Get Service Account

Get a service account

URI Template

```
GET /api/service-accounts/{name}
```

{name}

Service account name

Response Type

```
interface ServiceAccountInfo {  
  
    // Service account name  
    name: string;  
  
    // Displayed name  
    displayName: string;  
  
    // Whether the account may login  
    active: boolean;  
  
    // User that created this user account  
    createdBy: UserInfo;  
  
    // When this user was created  
    creationTime: string; // RFC 3339 timestamp  
  
    // When this account was first activated  
    confirmationTime: string; // RFC 3339 timestamp  
  
    // When this account last logged in  
    lastLoginTime: string; // RFC 3339 timestamp  
}
```

Related Types

```
interface UserInfo {  
  
    // Username  
    name: string;  
  
    // Displayed name  
    displayName: string;  
  
    // Email address  
    email: string;  
  
    // Whether the user may login  
    active: boolean;  
  
    // Whether the user has all privileges  
    superuser: boolean;  
  
    // User that created this user account  
    createdBy: UserInfo;  
  
    // When this user was created  
    creationTime: string; // RFC 3339 timestamp  
  
    // When this user was first activated
```

(continues on next page)

```

confirmationTime: string; // RFC 3339 timestamp

// When this user last logged in
lastLoginTime: string; // RFC 3339 timestamp

// System privileges
systemPrivileges: string[];
systemPrivilege: string[];

// Object privileges
objectPrivileges: ObjectPrivilegeInfo[];
objectPrivilege: ObjectPrivilegeInfo[];

// Groups that this user is member of
groups: GroupInfo[];

// External identities
identities: ExternalIdentityInfo[];

// Assigned roles
roles: RoleInfo[];

// Clearance level. If the command clearance feature is enabled,
// then this user attribute is used as an additional check whether
// the user may send certain commands.
//
// The command clearance feature is disabled by default.
clearance: SignificanceLevelType;
}

interface ObjectPrivilegeInfo {

    // Privilege type
    type: string;
    object: string[];

    // Objects of this type
    objects: string[];
}

interface GroupInfo {

    // Group name
    name: string;

    // Group description
    description: string;

    // Users that are member of this group
    users: UserInfo[];

    // Service accounts that are member of this group
    serviceAccounts: ServiceAccountInfo[];
}

interface ExternalIdentityInfo {

    // External identity
    identity: string;

    // Name of the identity provider
    provider: string;
}

interface RoleInfo {

    // Role name
    name: string;

    // Role description
    description: string;
}

```

(continued from previous page)

```
// System privileges
systemPrivileges: string[];

// Object privileges
objectPrivileges: ObjectPrivilegeInfo[];

// Whether this role is assigned by default
default: boolean;
}

enum SignificanceLevelType {
    NONE = "NONE",
    WATCH = "WATCH",
    WARNING = "WARNING",
    DISTRESS = "DISTRESS",
    CRITICAL = "CRITICAL",
    SEVERE = "SEVERE",
}
```

12.19 Delete Service Account

Delete a service account

URI Template

```
DELETE /api/service-accounts/{name}
```

{name}

Service account name

12.20 Create Service Account

Create a service account

URI Template

```
POST /api/service-accounts
```

Request Body

```
interface CreateServiceAccountRequest {

    // Service account name
    name: string;
}
```

Response Type

```
interface CreateServiceAccountResponse {  
    // Service account name  
    name: string;  
    applicationId: string;  
    applicationSecret: string;  
}
```


13. Indexes

13.1 List Command History Index

List command history index

URI Template

```
GET /api/archive/{instance}/command-index
```

{instance}

Yamcs instance name.

Query Parameters

mergeTime

Value in milliseconds that indicates the maximum gap before two consecutive index ranges are merged together. Default: 2000

limit

The maximum number of returned entries. Choose this value too high and you risk hitting the maximum response size limit enforced by the server. Default: 1000. Note that in general it is advised to control the size of the response via `mergeTime`, rather than via `limit`.

start

Filter the lower bound of the index entries. Specify a date string in ISO 8601 format.

stop

Filter the upper bound of the index entries. Specify a date string in ISO 8601 format.

next

Continuation token returned by a previous page response.

name

Filter on a specific command

Response Type

```
interface IndexResponse {
  group: IndexGroup[];

  // Token indicating the response is only partial. More results can then
  // be obtained by performing the same request (including all original
  // query parameters) and setting the ``next`` parameter to this token.
  continuationToken: string;
}
```

Related Types

```
interface IndexGroup {
  id: NamedObjectId;
  entry: IndexEntry[];
}

// Used by external clients to identify an item in the Mission Database
// If namespace is set, then the name is that of an alias, rather than
// the qualified name.
interface NamedObjectId {
  name: string;
  namespace: string;
}

interface IndexEntry {
  start: string;
  stop: string;
  count: number;
  seqStart: string; // String decimal
  seqStop: string; // String decimal
}
```

13.2 List Event Index

List event index

URI Template

```
GET /api/archive/{instance}/event-index
```

{instance}

Yamcs instance name.

Query Parameters

mergeTime

Value in milliseconds that indicates the maximum gap before two consecutive index ranges are merged together. Default: 2000

limit

The maximum number of returned entries. Choose this value too high and you risk hitting the maximum response size limit enforced by the server. Default: 1000. Note that in general it is advised to control the size of the response via `mergeTime`, rather than via `limit`.

start

Filter the lower bound of the index entries. Specify a date string in ISO 8601 format.

stop

Filter the upper bound of the index entries. Specify a date string in ISO 8601 format.

next

Continuation token returned by a previous page response.

source

Filter on specific sources.

Response Type

```
interface IndexResponse {
    group: IndexGroup[];

    // Token indicating the response is only partial. More results can then
    // be obtained by performing the same request (including all original
    // query parameters) and setting the ``next`` parameter to this token.
    continuationToken: string;
}
```

Related Types

```
interface IndexGroup {
    id: NamedObjectId;
    entry: IndexEntry[];
}

// Used by external clients to identify an item in the Mission Database
// If namespace is set, then the name is that of an alias, rather than
// the qualified name.
interface NamedObjectId {
    name: string;
    namespace: string;
}

interface IndexEntry {
    start: string;
    stop: string;
    count: number;
    seqStart: string; // String decimal
    seqStop: string; // String decimal
}
```

13.3 List Packet Index

List packet index

URI Template

```
GET /api/archive/{instance}/packet-index
```

{instance}

Yamcs instance name.

Query Parameters

mergeTime

Value in milliseconds that indicates the maximum gap before two consecutive index ranges are merged together. Default: 2000

limit

The maximum number of returned entries. Choose this value too high and you risk hitting the maximum response size limit enforced by the server. Default: 1000. Note that in general it is advised to control the size of the response via `mergeTime`, rather than via `limit`.

start

Filter the lower bound of the index entries. Specify a date string in ISO 8601 format.

stop

Filter the upper bound of the index entries. Specify a date string in ISO 8601 format.

next

Continuation token returned by a previous page response.

name

Filter on specific packet names.

Response Type

```
interface IndexResponse {  
    group: IndexGroup[];  
  
    // Token indicating the response is only partial. More results can then  
    // be obtained by performing the same request (including all original  
    // query parameters) and setting the ``next`` parameter to this token.  
    continuationToken: string;  
}
```

Related Types

```
interface IndexGroup {  
    id: NamedObjectId;  
    entry: IndexEntry[];  
}  
  
// Used by external clients to identify an item in the Mission Database  
// If namespace is set, then the name is that of an alias, rather than
```

(continues on next page)

(continued from previous page)

```
// the qualified name.
interface NamedObjectId {
    name: string;
    namespace: string;
}

interface IndexEntry {
    start: string;
    stop: string;
    count: number;
    seqStart: string; // String decimal
    seqStop: string; // String decimal
}
```

13.4 List Parameter Index

List parameter index

URI Template

```
GET /api/archive/{instance}/parameter-index
```

{instance}

Yamcs instance name.

Query Parameters

mergeTime

Value in milliseconds that indicates the maximum gap before two consecutive index ranges are merged together. Default: 20000

limit

The maximum number of returned entries. Choose this value too high and you risk hitting the maximum response size limit enforced by the server. Default: 1000. Note that in general it is advised to control the size of the response via `mergeTime`, rather than via `limit`.

start

Filter the lower bound of the index entries. Specify a date string in ISO 8601 format.

stop

Filter the upper bound of the index entries. Specify a date string in ISO 8601 format.

next

Continuation token returned by a previous page response.

group

Filter on specific parameter groups.

Response Type

```
interface IndexResponse {
  group: IndexGroup[];

  // Token indicating the response is only partial. More results can then
  // be obtained by performing the same request (including all original
  // query parameters) and setting the `next` parameter to this token.
  continuationToken: string;
}
```

Related Types

```
interface IndexGroup {
  id: NamedObjectId;
  entry: IndexEntry[];
}

// Used by external clients to identify an item in the Mission Database
// If namespace is set, then the name is that of an alias, rather than
// the qualified name.
interface NamedObjectId {
  name: string;
  namespace: string;
}

interface IndexEntry {
  start: string;
  stop: string;
  count: number;
  seqStart: string; // String decimal
  seqStop: string; // String decimal
}
```

13.5 List Completeness Index

List completeness index

URI Template

```
GET /api/archive/{instance}/completeness-index
```

{instance}

Yamcs instance name.

Query Parameters

limit

The maximum number of returned entries. Choose this value too high and you risk hitting the maximum response size limit enforced by the server. Default: 1000. Note that in general it is advised to control the size of the response via `mergeTime`, rather than via `limit`.

start

Filter the lower bound of the index entries. Specify a date string in ISO 8601 format.

stop

Filter the upper bound of the index entries. Specify a date string in ISO 8601 format.

next

Continuation token returned by a previous page response.

Response Type

```
interface IndexResponse {
  group: IndexGroup[];

  // Token indicating the response is only partial. More results can then
  // be obtained by performing the same request (including all original
  // query parameters) and setting the ``next`` parameter to this token.
  continuationToken: string;
}
```

Related Types

```
interface IndexGroup {
  id: NamedObjectId;
  entry: IndexEntry[];
}

// Used by external clients to identify an item in the Mission Database
// If namespace is set, then the name is that of an alias, rather than
// the qualified name.
interface NamedObjectId {
  name: string;
  namespace: string;
}

interface IndexEntry {
  start: string;
  stop: string;
  count: number;
  seqStart: string; // String decimal
  seqStop: string; // String decimal
}
```

13.6 Stream Packet Index

Streams back packet index records

Warning: This method uses server-streaming. Yamcs sends an unspecified amount of data using chunked transfer encoding.

URI Template

```
POST /api/archive/{instance}:streamPacketIndex
```

{instance}

Yamcs instance name.

Request Body

```
interface StreamPacketIndexRequest {  
  
    // The time at which to start retrieving index records.  
    start: string; // RFC 3339 timestamp  
  
    // The time at which to stop retrieving index records.  
    stop: string; // RFC 3339 timestamp  
    names: string[];  
}
```

Response Type

```
//contains histogram data  
interface ArchiveRecord {  
    id: NamedObjectId;  
    num: number;  
    seqFirst: string; // String decimal  
    seqLast: string; // String decimal  
    first: string; // RFC 3339 timestamp  
    last: string; // RFC 3339 timestamp  
    extra: {[key: string]: string};  
}
```

Related Types

```
// Used by external clients to identify an item in the Mission Database  
// If namespace is set, then the name is that of an alias, rather than  
// the qualified name.  
interface NamedObjectId {  
    name: string;  
    namespace: string;  
}
```

13.7 Stream Parameter Index

Streams back parameter index records

Warning: This method uses server-streaming. Yamcs sends an unspecified amount of data using chunked transfer encoding.

URI Template

```
POST /api/archive/{instance}:streamParameterIndex
```

{instance}
Yamcs instance name.

Request Body


```
interface StreamParameterIndexRequest {

    // The time at which to start retrieving index records.
    start: string; // RFC 3339 timestamp

    // The time at which to stop retrieving index records.
    stop: string; // RFC 3339 timestamp
}
```

Response Type

```
//contains histogram data
interface ArchiveRecord {
    id: NamedObjectId;
    num: number;
    seqFirst: string; // String decimal
    seqLast: string; // String decimal
    first: string; // RFC 3339 timestamp
    last: string; // RFC 3339 timestamp
    extra: {[key: string]: string};
}
```

Related Types

```
// Used by external clients to identify an item in the Mission Database
// If namespace is set, then the name is that of an alias, rather than
// the qualified name.
interface NamedObjectId {
    name: string;
    namespace: string;
}
```

13.8 Stream Command Index

Streams back processed parameter index records

Warning: This method uses server-streaming. Yamcs sends an unspecified amount of data using chunked transfer encoding.

URI Template

```
POST /api/archive/{instance}:streamCommandIndex
```

{instance}
Yamcs instance name.

Request Body

```
interface StreamCommandIndexRequest {

    // The time at which to start retrieving index records.
    start: string; // RFC 3339 timestamp

    // The time at which to stop retrieving index records.
```

(continues on next page)

(continued from previous page)

```
    stop: string; // RFC 3339 timestamp
}
```

Response Type

```
//contains histogram data
interface ArchiveRecord {
    id: NamedObjectId;
    num: number;
    seqFirst: string; // String decimal
    seqLast: string; // String decimal
    first: string; // RFC 3339 timestamp
    last: string; // RFC 3339 timestamp
    extra: {[key: string]: string};
}
```

Related Types

```
// Used by external clients to identify an item in the Mission Database
// If namespace is set, then the name is that of an alias, rather than
// the qualified name.
interface NamedObjectId {
    name: string;
    namespace: string;
}
```

13.9 Stream Event Index

Streams back event index records

Warning: This method uses server-streaming. Yamcs sends an unspecified amount of data using chunked transfer encoding.

URI Template

```
POST /api/archive/{instance}:streamEventIndex
```

{instance}
Yamcs instance name.

Request Body

```
interface StreamEventIndexRequest {

    // The time at which to start retrieving index records.
    start: string; // RFC 3339 timestamp

    // The time at which to stop retrieving index records.
    stop: string; // RFC 3339 timestamp
}
```

Response Type

```
//contains histogram data
interface ArchiveRecord {
  id: NamedObjectId;
  num: number;
  seqFirst: string; // String decimal
  seqLast: string; // String decimal
  first: string; // RFC 3339 timestamp
  last: string; // RFC 3339 timestamp
  extra: {[key: string]: string};
}
```

Related Types

```
// Used by external clients to identify an item in the Mission Database
// If namespace is set, then the name is that of an alias, rather than
// the qualified name.
interface NamedObjectId {
  name: string;
  namespace: string;
}
```

13.10 Stream Completeness Index

Streams back event index records

Warning: This method uses server-streaming. Yamcs sends an unspecified amount of data using chunked transfer encoding.

URI Template

```
POST /api/archive/{instance}:streamCompletenessIndex
```

{instance}

Yamcs instance name.

Request Body

```
interface StreamCompletenessIndexRequest {

  // The time at which to start retrieving index records.
  start: string; // RFC 3339 timestamp

  // The time at which to stop retrieving index records.
  stop: string; // RFC 3339 timestamp
}
```

Response Type

```
//contains histogram data
interface ArchiveRecord {
  id: NamedObjectId;
  num: number;
  seqFirst: string; // String decimal
  seqLast: string; // String decimal
  first: string; // RFC 3339 timestamp
  last: string; // RFC 3339 timestamp
  extra: {[key: string]: string};
}
```

Related Types

```
// Used by external clients to identify an item in the Mission Database
// If namespace is set, then the name is that of an alias, rather than
// the qualified name.
interface NamedObjectId {
  name: string;
  namespace: string;
}
```

13.11 Rebuild Ccsds Index

Rebuild CCSDS TM Index

URI Template

```
POST /api/archive/{instance}:rebuildCcsdsIndex
```

{instance}
Yamcs instance name.

Request Body

```
interface RebuildCcsdsIndexRequest {
  start: string; // RFC 3339 timestamp
  stop: string; // RFC 3339 timestamp
}
```

14. Links

14.1 List Links

List links

URI Template

```
GET /api/links/{instance?}
```

{instance?}
Yamcs instance name.

Response Type

```
interface ListLinksResponse {  
    links: LinkInfo[];  
}
```

Related Types

```
interface LinkInfo {  
    instance: string;  
    name: string;  
    type: string;  
    spec: string;  
    disabled: boolean;  
    status: string;  
    dataInCount: string; // String decimal  
    dataOutCount: string; // String decimal  
    detailedStatus: string;  
  
    //if this is a sublink of an aggregated data link, this is the name of the parent  
    parentName: string;  
}
```

14.2 Get Link

Get a link

URI Template

```
GET /api/links/{instance}/{link}
```

{instance}

Yamcs instance name.

{link}

Link name.

Response Type

```
interface LinkInfo {
  instance: string;
  name: string;
  type: string;
  spec: string;
  disabled: boolean;
  status: string;
  dataInCount: string; // String decimal
  dataOutCount: string; // String decimal
  detailedStatus: string;

  //if this is a sublink of an aggregated data link, this is the name of the parent
  parentName: string;
}
```

14.3 Update Link

Update a link

URI Template

```
PATCH /api/links/{instance}/{link}
```

{instance}

Yamcs instance name.

{link}

Link name.

Request Body

```
interface EditLinkRequest {

  // The state of the link. Either ``enabled`` or ``disabled``.
  state: string;
  resetCounters: boolean;
}
```

Response Type

```
interface LinkInfo {
  instance: string;
  name: string;
  type: string;
  spec: string;
  disabled: boolean;
```

(continues on next page)

(continued from previous page)

```
status: string;
dataInCount: string; // String decimal
dataOutCount: string; // String decimal
detailedStatus: string;

//if this is a sublink of an aggregated data link, this is the name of the parent
parentName: string;
}
```

14.4 Enable Link

Enable a link

URI Template

```
POST /api/links/{instance}/{link}:enable
```

{instance}
Yamcs instance name.

{link}
Link name.

Response Type

```
interface LinkInfo {
  instance: string;
  name: string;
  type: string;
  spec: string;
  disabled: boolean;
  status: string;
  dataInCount: string; // String decimal
  dataOutCount: string; // String decimal
  detailedStatus: string;

  //if this is a sublink of an aggregated data link, this is the name of the parent
  parentName: string;
}
```

14.5 Disable Link

Disable a link

URI Template

```
POST /api/links/{instance}/{link}:disable
```

{instance}
Yamcs instance name.

{link}
Link name.

Response Type

```
interface LinkInfo {
  instance: string;
  name: string;
  type: string;
  spec: string;
  disabled: boolean;
  status: string;
  dataInCount: string; // String decimal
  dataOutCount: string; // String decimal
  detailedStatus: string;

  //if this is a sublink of an aggregated data link, this is the name of the parent
  parentName: string;
}
```

14.6 Reset Link Counters

Reset link counters

URI Template

```
POST /api/links/{instance}/{link}:resetCounters
```

{instance}
Yamcs instance name.

{link}
Link name.

Response Type

```
interface LinkInfo {
  instance: string;
  name: string;
  type: string;
  spec: string;
  disabled: boolean;
  status: string;
  dataInCount: string; // String decimal
  dataOutCount: string; // String decimal
  detailedStatus: string;

  //if this is a sublink of an aggregated data link, this is the name of the parent
  parentName: string;
}
```

14.7 Subscribe Links

Receive link updates

WebSocket

This method requires to upgrade an HTTP connection to WebSocket. See details on [how Yamcs uses WebSocket](https://docs.yamcs.org/yamcs-http-api/websocket)¹⁰.

Use the message type links.

Input Type

```
interface SubscribeLinksRequest {  
    instance: string;  
}
```

Output Type

```
interface LinkEvent {  
    type: Type;  
    linkInfo: LinkInfo;  
}
```

Related Types

```
interface LinkInfo {  
    instance: string;  
    name: string;  
    type: string;  
    spec: string;  
    disabled: boolean;  
    status: string;  
    dataInCount: string; // String decimal  
    dataOutCount: string; // String decimal  
    detailedStatus: string;  
  
    //if this is a sublink of an aggregated data link, this is the name of the parent  
    parentName: string;  
}  
  
enum Type {  
  
    // A new link was registered. You also receive this event directly after you subscribe,  
    // for every link that is registered at that time.  
    REGISTERED = "REGISTERED",  
  
    // A link was unregistered.  
    UNREGISTERED = "UNREGISTERED",  
  
    // A link was updated in one of its attributes, for example the dataCount has increased,  
    // or the status has changed.  
    UPDATED = "UPDATED",  
}
```

¹⁰ <https://docs.yamcs.org/yamcs-http-api/websocket>

15. Management

15.1 Get System Info

Get system info

URI Template

```
GET /api/sysinfo
```

Response Type

```
interface SystemInfo {  
  
    // Yamcs version number.  
    yamcsVersion: string;  
  
    // Git revision number at build time.  
    revision: string;  
  
    // Server identifier, as used in system parameters and distributed setups.  
    serverId: string;  
  
    // Uptime of Yamcs in milliseconds  
    uptime: string; // String decimal  
  
    // Java virtual machine implementation name, version and vendor  
    jvm: string;  
  
    // Working directory of Yamcs (base path for relative references)  
    workingDirectory: string;  
  
    // Directory where configuration files are located.  
    configDirectory: string;  
  
    // Directory where Yamcs data is stored.  
    dataDirectory: string;  
  
    // Directory where Yamcs may cache files.  
    cacheDirectory: string;  
  
    // Operating system name and version.  
    os: string;  
  
    // Operating system architecture.  
    arch: string;  
  
    // Number of processors available to Yamcs.  
    availableProcessors: number;  
  
    // System load average for the last minute.  
    loadAverage: number;  
}
```

(continues on next page)

(continued from previous page)

```
// Amount of memory in bytes of the heap that is used for object allocation and that is committed for
↳ the JVM to use.
heapMemory: string; // String decimal

// Amount of used heap memory in bytes.
usedHeapMemory: string; // String decimal

// Maximum amount of heap memory in bytes that can be used for memory management.
maxHeapMemory: string; // String decimal

// Amount of non-heap memory in bytes that is committed for the JVM to use.
nonHeapMemory: string; // String decimal

// Amount of non-heap used memory in bytes.
usedNonHeapMemory: string; // String decimal

// Maximum amount of non-heap memory in bytes that can be used for memory management.
maxNonHeapMemory: string; // String decimal

// Number of active threads.
jvmThreadCount: string; // String decimal

// Root directories, each corresponding to a distinct file hierarchy.
rootDirectories: RootDirectory[];

// Information about the Yamcs process and any descendants.
process: ProcessInfo;
}
```

Related Types

```
interface RootDirectory {

    // Root directory location.
    directory: string;

    // The type of the file store where this root directory is located.
    type: string;

    // Size in bytes of the file store where this root directory is located.
    totalSpace: string; // String decimal

    // Number of unallocated bytes in the file store where this root directory
    // is located.
    unallocatedSpace: string; // String decimal

    // Number of bytes available to Yamcs on the file store where this root
    // directory is located.
    usableSpace: string; // String decimal
}

interface ProcessInfo {

    // Native process ID.
    pid: string; // String decimal

    // User of the process.
    user: string;

    // Executable pathname of the process.
    command: string;

    // Arguments of the process.
    arguments: string[];

    // Start time of the process.
    startTime: string; // RFC 3339 timestamp
}
```

(continues on next page)

(continued from previous page)

```
// Accumulated total cputime.
totalCpuDuration: string; // Duration in seconds. Example: "3s" or "3.001s"

// Direct children of the process.
children: ProcessInfo[];
}
```

15.2 Subscribe System Info

Receive system info updates

WebSocket

This method requires to upgrade an HTTP connection to WebSocket. See details on [how Yamcs uses WebSocket¹¹](#).

Use the message type sysinfo.

Output Type

```
interface SystemInfo {

    // Yamcs version number.
    yamcsVersion: string;

    // Git revision number at build time.
    revision: string;

    // Server identifier, as used in system parameters and distributed setups.
    serverId: string;

    // Uptime of Yamcs in milliseconds
    uptime: string; // String decimal

    // Java virtual machine implementation name, version and vendor
    jvm: string;

    // Working directory of Yamcs (base path for relative references)
    workingDirectory: string;

    // Directory where configuration files are located.
    configDirectory: string;

    // Directory where Yamcs data is stored.
    dataDirectory: string;

    // Directory where Yamcs may cache files.
    cacheDirectory: string;

    // Operating system name and version.
    os: string;

    // Operating system architecture.
    arch: string;

    // Number of processors available to Yamcs.
    availableProcessors: number;

    // System load average for the last minute.
    loadAverage: number;
}
```

(continues on next page)

¹¹ <https://docs.yamcs.org/yamcs-http-api/websocket>

```

// Amount of memory in bytes of the heap that is used for object allocation and that is committed for
the JVM to use.
heapMemory: string; // String decimal

// Amount of used heap memory in bytes.
usedHeapMemory: string; // String decimal

// Maximum amount of heap memory in bytes that can be used for memory management.
maxHeapMemory: string; // String decimal

// Amount of non-heap memory in bytes that is committed for the JVM to use.
nonHeapMemory: string; // String decimal

// Amount of non-heap used memory in bytes.
usedNonHeapMemory: string; // String decimal

// Maximum amount of non-heap memory in bytes that can be used for memory management.
maxNonHeapMemory: string; // String decimal

// Number of active threads.
jvmThreadCount: string; // String decimal

// Root directories, each corresponding to a distinct file hierarchy.
rootDirectories: RootDirectory[];

// Information about the Yamcs process and any descendants.
process: ProcessInfo;
}

```

Related Types

```

interface RootDirectory {

// Root directory location.
directory: string;

// The type of the file store where this root directory is located.
type: string;

// Size in bytes of the file store where this root directory is located.
totalSpace: string; // String decimal

// Number of unallocated bytes in the file store where this root directory
// is located.
unallocatedSpace: string; // String decimal

// Number of bytes available to Yamcs on the file store where this root
// directory is located.
usableSpace: string; // String decimal
}

```

```

interface ProcessInfo {

// Native process ID.
pid: string; // String decimal

// User of the process.
user: string;

// Executable pathname of the process.
command: string;

// Arguments of the process.
arguments: string[];

// Start time of the process.
startTime: string; // RFC 3339 timestamp
}

```

(continues on next page)

```
// Accumulated total cputime.
totalCpuDuration: string; // Duration in seconds. Example: "3s" or "3.001s"

// Direct children of the process.
children: ProcessInfo[];
}
```

15.3 List Instance Templates

List instance templates

URI Template

```
GET /api/instance-templates
```

Response Type

```
interface ListInstanceTemplatesResponse {
  templates: InstanceTemplate[];
}
```

Related Types

```
interface InstanceTemplate {

  // Template name.
  name: string;

  // Human-friendly description
  description: string;

  // List of variables that this template may expect
  variables: TemplateVariable[];
}

interface TemplateVariable {

  // Variable name.
  name: string;

  // Verbose name for use in UI forms
  label: string;

  // Type of variable (Java class extending org.yamcs.templating.Variable)
  type: string;

  // Verbose user guidance (HTML)
  help: string;

  // Whether this variable is required input
  required: boolean;

  // List of valid choices
  choices: string[];

  // Initial value for use in UI forms
  initial: string;
}
```

15.4 Get Instance Template

Get an instance template

URI Template

```
GET /api/instance-templates/{template}
```

{template}
Template name.

Response Type

```
interface InstanceTemplate {  
  
    // Template name.  
    name: string;  
  
    // Human-friendly description  
    description: string;  
  
    // List of variables that this template may expect  
    variables: TemplateVariable[];  
}
```

Related Types

```
interface TemplateVariable {  
  
    // Variable name.  
    name: string;  
  
    // Verbose name for use in UI forms  
    label: string;  
  
    // Type of variable (Java class extending org.yamcs.templating.Variable)  
    type: string;  
  
    // Verbose user guidance (HTML)  
    help: string;  
  
    // Whether this variable is required input  
    required: boolean;  
  
    // List of valid choices  
    choices: string[];  
  
    // Initial value for use in UI forms  
    initial: string;  
}
```

15.5 List Instances

List instances

URI Template

```
GET /api/instances
```

Query Parameters

filter

Response Type

```
interface ListInstancesResponse {  
    instances: YamcsInstance[];  
}
```

Related Types

```
interface YamcsInstance {  
  
    // Instance name.  
    name: string;  
    missionDatabase: MissionDatabase;  
    processors: ProcessorInfo[];  
    state: InstanceState;  
  
    //in case the state=FAILED, this field will indicate the cause of the failure  
    // the missionDatabase and other fields may not be filled when this happens  
    failureCause: string;  
    missionTime: string; // RFC 3339 timestamp  
  
    // Labels assigned to this instance. Each entry is keyed by the tag name  
    // of the label. The value represent the label value for that tag.  
    labels: {[key: string]: string};  
  
    // Feature capability hints for client use  
    capabilities: string[];  
  
    // Name of the template, if this instance was generated  
    template: string;  
  
    // Arguments used during template processing, if this instance  
    // was generated  
    templateArgs: {[key: string]: string};  
  
    // Whether the template is stil available  
    templateAvailable: boolean;  
  
    // Whether the template has changed since this instance was  
    // generated  
    templateChanged: boolean;  
}
```

```
interface MissionDatabase {  
  
    // This is the config section in mdb.yaml  
    configName: string;  
  
    // Root space-system name  
    name: string;  
  
    // Root space-system header version  
    version: string;  
    spaceSystem: SpaceSystemInfo[];
```

(continues on next page)


```

parameterCount: number;
containerCount: number;
commandCount: number;
algorithmCount: number;
parameterTypeCount: number;
}

interface SpaceSystemInfo {
  name: string;
  qualifiedName: string;
  shortDescription: string;
  longDescription: string;
  alias: NamedObjectId[];
  version: string;
  history: HistoryInfo[];
  sub: SpaceSystemInfo[];
  ancillaryData: {[key: string]: AncillaryDataInfo};
}

// Used by external clients to identify an item in the Mission Database
// If namespace is set, then the name is that of an alias, rather than
// the qualified name.
interface NamedObjectId {
  name: string;
  namespace: string;
}

interface HistoryInfo {
  version: string;
  date: string;
  message: string;
  author: string;
}

interface ProcessorInfo {

  // Yamcs instance name.
  instance: string;

  // Processor name.
  name: string;
  type: string;
  spec: string;
  creator: string;
  hasAlarms: boolean;
  hasCommanding: boolean;
  state: ServiceState;
  replayRequest: ReplayRequest;
  replayState: ReplayState;
  services: ServiceInfo[];
  persistent: boolean;
  time: string; // RFC 3339 timestamp
  replay: boolean;
  checkCommandClearance: boolean;
}

//used to replay (concurrently) TM packets, parameters and events
interface ReplayRequest {

  // **Required.** The time at which the replay should start.
  start: string; // RFC 3339 timestamp

  // The time at which the replay should stop.
  // If unspecified, the replay will keep going as long as there is remaining data.
  stop: string; // RFC 3339 timestamp

  //what should happen at the end of the replay
  endAction: EndAction;

  //how fast the replay should go
  speed: ReplaySpeed;
}

```

```

// Reverse the direction of the replay
reverse: boolean;
parameterRequest: ParameterReplayRequest;

// By default all Packets, Events, CommandHistory are part of the replay
// Unless one or more of the below requests are specified.
packetRequest: PacketReplayRequest;
eventRequest: EventReplayRequest;
commandHistoryRequest: CommandHistoryReplayRequest;
ppRequest: PpReplayRequest;
}

interface ReplaySpeed {
    type: ReplaySpeedType;
    param: number;
}

interface ParameterReplayRequest {
    nameFilter: NamedObjectId[];
    sendRaw: boolean;
    performMonitoring: boolean;
}

interface PacketReplayRequest {

    // No filter, means all packets for which privileges exist, are sent
    nameFilter: NamedObjectId[];
}

interface EventReplayRequest {
}

interface CommandHistoryReplayRequest {

    // No filter, means all command history entries are sent
    nameFilter: NamedObjectId[];
}

//Request to replay parameters - they can be filtered by the parameter group
interface PpReplayRequest {

    // No filter, means all pp groups are sent
    groupNameFilter: string[];

    // exclude the parameters from these groups
    // this takes precedence over the filter above (i.e. if a group is part of both, it will be excluded)
    groupNameExclude: string[];
}

interface ServiceInfo {
    instance: string;
    name: string;
    state: ServiceState;
    className: string;
    processor: string;
}

enum ServiceState {
    NEW = "NEW",
    STARTING = "STARTING",
    RUNNING = "RUNNING",
    STOPPING = "STOPPING",
    TERMINATED = "TERMINATED",
    FAILED = "FAILED",
}

enum EndAction {
    LOOP = "LOOP",
    QUIT = "QUIT",
    STOP = "STOP",
}

```

```

}

enum ReplaySpeedType {
    AFAP = "AFAP",
    FIXED_DELAY = "FIXED_DELAY",
    REALTIME = "REALTIME",
    STEP_BY_STEP = "STEP_BY_STEP",
}

enum ReplayState {

    // just at the beginning or when the replay request (start, stop or packet selection) changes
    INITIALIZATION = "INITIALIZATION",
    RUNNING = "RUNNING",

    // The replay has reached the end with the endaction stop
    STOPPED = "STOPPED",

    // The replay stopped due to an error.
    ERROR = "ERROR",
    PAUSED = "PAUSED",

    // The replay is finished and closed
    CLOSED = "CLOSED",
}

enum ServiceState {
    NEW = "NEW",
    STARTING = "STARTING",
    RUNNING = "RUNNING",
    STOPPING = "STOPPING",
    TERMINATED = "TERMINATED",
    FAILED = "FAILED",
}

enum InstanceState {
    OFFLINE = "OFFLINE",
    INITIALIZING = "INITIALIZING",
    INITIALIZED = "INITIALIZED",
    STARTING = "STARTING",
    RUNNING = "RUNNING",
    STOPPING = "STOPPING",
    FAILED = "FAILED",
}

```

15.6 Subscribe Instances

Receive instance updates

WebSocket

This method requires to upgrade an HTTP connection to WebSocket. See details on [how Yamcs uses WebSocket](https://docs.yamcs.org/yamcs-http-api/websocket)¹².

Use the message type instances.

Output Type

¹² <https://docs.yamcs.org/yamcs-http-api/websocket>

```

interface YamcsInstance {

    // Instance name.
    name: string;
    missionDatabase: MissionDatabase;
    processors: ProcessorInfo[];
    state: InstanceState;

    //in case the state=FAILED, this field will indicate the cause of the failure
    // the missionDatabase and other fields may not be filled when this happens
    failureCause: string;
    missionTime: string; // RFC 3339 timestamp

    // Labels assigned to this instance. Each entry is keyed by the tag name
    // of the label. The value represent the label value for that tag.
    labels: {[key: string]: string};

    // Feature capability hints for client use
    capabilities: string[];

    // Name of the template, if this instance was generated
    template: string;

    // Arguments used during template processing, if this instance
    // was generated
    templateArgs: {[key: string]: string};

    // Whether the template is stil available
    templateAvailable: boolean;

    // Whether the template has changed since this instance was
    // generated
    templateChanged: boolean;
}

```

Related Types

```

interface MissionDatabase {

    // This is the config section in mdb.yaml
    configName: string;

    // Root space-system name
    name: string;

    // Root space-system header version
    version: string;
    spaceSystem: SpaceSystemInfo[];
    parameterCount: number;
    containerCount: number;
    commandCount: number;
    algorithmCount: number;
    parameterTypeCount: number;
}

interface SpaceSystemInfo {
    name: string;
    qualifiedName: string;
    shortDescription: string;
    longDescription: string;
    alias: NamedObjectId[];
    version: string;
    history: HistoryInfo[];
    sub: SpaceSystemInfo[];
    ancillaryData: {[key: string]: AncillaryDataInfo};
}

// Used by external clients to identify an item in the Mission Database
// If namespace is set, then the name is that of an alias, rather than

```

(continues on next page)

```

// the qualified name.
interface NamedObjectId {
    name: string;
    namespace: string;
}

interface HistoryInfo {
    version: string;
    date: string;
    message: string;
    author: string;
}

interface ProcessorInfo {

    // Yamcs instance name.
    instance: string;

    // Processor name.
    name: string;
    type: string;
    spec: string;
    creator: string;
    hasAlarms: boolean;
    hasCommanding: boolean;
    state: ServiceState;
    replayRequest: ReplayRequest;
    replayState: ReplayState;
    services: ServiceInfo[];
    persistent: boolean;
    time: string; // RFC 3339 timestamp
    replay: boolean;
    checkCommandClearance: boolean;
}

//used to replay (concurrently) TM packets, parameters and events
interface ReplayRequest {

    // **Required.** The time at which the replay should start.
    start: string; // RFC 3339 timestamp

    // The time at which the replay should stop.
    // If unspecified, the replay will keep going as long as there is remaining data.
    stop: string; // RFC 3339 timestamp

    //what should happen at the end of the replay
    endAction: EndAction;

    //how fast the replay should go
    speed: ReplaySpeed;

    // Reverse the direction of the replay
    reverse: boolean;
    parameterRequest: ParameterReplayRequest;

    // By default all Packets, Events, CommandHistory are part of the replay
    // Unless one or more of the below requests are specified.
    packetRequest: PacketReplayRequest;
    eventRequest: EventReplayRequest;
    commandHistoryRequest: CommandHistoryReplayRequest;
    ppRequest: PpReplayRequest;
}

interface ReplaySpeed {
    type: ReplaySpeedType;
    param: number;
}

interface ParameterReplayRequest {
    nameFilter: NamedObjectId[];
    sendRaw: boolean;
}

```

```

    performMonitoring: boolean;
}

interface PacketReplayRequest {

    // No filter, means all packets for which privileges exist, are sent
    nameFilter: NamedObjectId[];
}

interface EventReplayRequest {
}

interface CommandHistoryReplayRequest {

    // No filter, means all command history entries are sent
    nameFilter: NamedObjectId[];
}

//Request to replay parameters - they can be filtered by the parameter group
interface PpReplayRequest {

    // No filter, means all pp groups are sent
    groupNameFilter: string[];

    // exclude the parameters from these groups
    // this takes precedence over the filter above (i.e. if a group is part of both, it will be excluded)
    groupNameExclude: string[];
}

interface ServiceInfo {
    instance: string;
    name: string;
    state: ServiceState;
    className: string;
    processor: string;
}

enum ServiceState {
    NEW = "NEW",
    STARTING = "STARTING",
    RUNNING = "RUNNING",
    STOPPING = "STOPPING",
    TERMINATED = "TERMINATED",
    FAILED = "FAILED",
}

enum EndAction {
    LOOP = "LOOP",
    QUIT = "QUIT",
    STOP = "STOP",
}

enum ReplaySpeedType {
    AFAP = "AFAP",
    FIXED_DELAY = "FIXED_DELAY",
    REALTIME = "REALTIME",
    STEP_BY_STEP = "STEP_BY_STEP",
}

enum ReplayState {

    // just at the beginning or when the replay request (start, stop or packet selection) changes
    INITIALIZATION = "INITIALIZATION",
    RUNNING = "RUNNING",

    // The replay has reached the end with the endaction stop
    STOPPED = "STOPPED",

    // The replay stopped due to an error.
    ERROR = "ERROR",
    PAUSED = "PAUSED",
}

```

```

// The replay is finished and closed
CLOSED = "CLOSED",
}

enum ServiceState {
    NEW = "NEW",
    STARTING = "STARTING",
    RUNNING = "RUNNING",
    STOPPING = "STOPPING",
    TERMINATED = "TERMINATED",
    FAILED = "FAILED",
}

enum InstanceState {
    OFFLINE = "OFFLINE",
    INITIALIZING = "INITIALIZING",
    INITIALIZED = "INITIALIZED",
    STARTING = "STARTING",
    RUNNING = "RUNNING",
    STOPPING = "STOPPING",
    FAILED = "FAILED",
}

```

15.7 Get Instance

Get an instance

If an instance does not have web services enabled, it will be listed among the results, but none of its URLs will be filled in.

URI Template

```
GET /api/instances/{instance}
```

{instance}

Yamcs instance name.

Response Type

```

interface YamcsInstance {

    // Instance name.
    name: string;
    missionDatabase: MissionDatabase;
    processors: ProcessorInfo[];
    state: InstanceState;

    //in case the state=FAILED, this field will indicate the cause of the failure
    // the missionDatabase and other fields may not be filled when this happens
    failureCause: string;
    missionTime: string; // RFC 3339 timestamp

    // Labels assigned to this instance. Each entry is keyed by the tag name
    // of the label. The value represent the label value for that tag.
    labels: {[key: string]: string};

    // Feature capability hints for client use
    capabilities: string[];

    // Name of the template, if this instance was generated

```

(continues on next page)

(continued from previous page)

```
template: string;  
  
// Arguments used during template processing, if this instance  
// was generated  
templateArgs: {[key: string]: string};  
  
// Whether the template is still available  
templateAvailable: boolean;  
  
// Whether the template has changed since this instance was  
// generated  
templateChanged: boolean;  
}
```

Related Types

```
interface MissionDatabase {  
  
    // This is the config section in mdb.yaml  
    configName: string;  
  
    // Root space-system name  
    name: string;  
  
    // Root space-system header version  
    version: string;  
    spaceSystem: SpaceSystemInfo[];  
    parameterCount: number;  
    containerCount: number;  
    commandCount: number;  
    algorithmCount: number;  
    parameterTypeCount: number;  
}  
  
interface SpaceSystemInfo {  
    name: string;  
    qualifiedName: string;  
    shortDescription: string;  
    longDescription: string;  
    alias: NamedObjectId[];  
    version: string;  
    history: HistoryInfo[];  
    sub: SpaceSystemInfo[];  
    ancillaryData: {[key: string]: AncillaryDataInfo};  
}  
  
// Used by external clients to identify an item in the Mission Database  
// If namespace is set, then the name is that of an alias, rather than  
// the qualified name.  
interface NamedObjectId {  
    name: string;  
    namespace: string;  
}  
  
interface HistoryInfo {  
    version: string;  
    date: string;  
    message: string;  
    author: string;  
}  
  
interface ProcessorInfo {  
  
    // Yamcs instance name.  
    instance: string;  
  
    // Processor name.  
    name: string;
```

(continues on next page)


```

type: string;
spec: string;
creator: string;
hasAlarms: boolean;
hasCommanding: boolean;
state: ServiceState;
replayRequest: ReplayRequest;
replayState: ReplayState;
services: ServiceInfo[];
persistent: boolean;
time: string; // RFC 3339 timestamp
replay: boolean;
checkCommandClearance: boolean;
}

//used to replay (concurrently) TM packets, parameters and events
interface ReplayRequest {

    // **Required.** The time at which the replay should start.
    start: string; // RFC 3339 timestamp

    // The time at which the replay should stop.
    // If unspecified, the replay will keep going as long as there is remaining data.
    stop: string; // RFC 3339 timestamp

    //what should happen at the end of the replay
    endAction: EndAction;

    //how fast the replay should go
    speed: ReplaySpeed;

    // Reverse the direction of the replay
    reverse: boolean;
    parameterRequest: ParameterReplayRequest;

    // By default all Packets, Events, CommandHistory are part of the replay
    // Unless one or more of the below requests are specified.
    packetRequest: PacketReplayRequest;
    eventRequest: EventReplayRequest;
    commandHistoryRequest: CommandHistoryReplayRequest;
    ppRequest: PpReplayRequest;
}

interface ReplaySpeed {
    type: ReplaySpeedType;
    param: number;
}

interface ParameterReplayRequest {
    nameFilter: NamedObjectId[];
    sendRaw: boolean;
    performMonitoring: boolean;
}

interface PacketReplayRequest {

    // No filter, means all packets for which privileges exist, are sent
    nameFilter: NamedObjectId[];
}

interface EventReplayRequest {
}

interface CommandHistoryReplayRequest {

    // No filter, means all command history entries are sent
    nameFilter: NamedObjectId[];
}

//Request to replay parameters - they can be filtered by the parameter group
interface PpReplayRequest {

```

```

// No filter, means all pp groups are sent
groupNameFilter: string[];

// exclude the parameters from these groups
// this takes precedence over the filter above (i.e. if a group is part of both, it will be excluded)
groupNameExclude: string[];
}

interface ServiceInfo {
    instance: string;
    name: string;
    state: ServiceState;
    className: string;
    processor: string;
}

enum ServiceState {
    NEW = "NEW",
    STARTING = "STARTING",
    RUNNING = "RUNNING",
    STOPPING = "STOPPING",
    TERMINATED = "TERMINATED",
    FAILED = "FAILED",
}

enum EndAction {
    LOOP = "LOOP",
    QUIT = "QUIT",
    STOP = "STOP",
}

enum ReplaySpeedType {
    AFAP = "AFAP",
    FIXED_DELAY = "FIXED_DELAY",
    REALTIME = "REALTIME",
    STEP_BY_STEP = "STEP_BY_STEP",
}

enum ReplayState {

    // just at the beginning or when the replay request (start, stop or packet selection) changes
    INITIALIZATION = "INITIALIZATION",
    RUNNING = "RUNNING",

    // The replay has reached the end with the endaction stop
    STOPPED = "STOPPED",

    // The replay stopped due to an error.
    ERROR = "ERROR",
    PAUSED = "PAUSED",

    // The replay is finished and closed
    CLOSED = "CLOSED",
}

enum ServiceState {
    NEW = "NEW",
    STARTING = "STARTING",
    RUNNING = "RUNNING",
    STOPPING = "STOPPING",
    TERMINATED = "TERMINATED",
    FAILED = "FAILED",
}

enum InstanceState {
    OFFLINE = "OFFLINE",
    INITIALIZING = "INITIALIZING",
    INITIALIZED = "INITIALIZED",
    STARTING = "STARTING",
    RUNNING = "RUNNING",
}

```

(continued from previous page)

```
    STOPPING = "STOPPING",  
    FAILED = "FAILED",  
}
```

15.8 Create Instance

Create an instance

URI Template

POST /api/instances

Request Body

```
interface CreateInstanceRequest {  
  
    // **Required.** The name of the instance.  
    name: string;  
  
    // **Required.** The name of the template for this instance.  
    template: string;  
  
    // Arguments for substitution in the template definition. Each entry is  
    // keyed by the argument name. The value must be a string.  
    templateArgs: {[key: string]: string};  
  
    // Labels assigned to this instance. Each entry is keyed by the tag name  
    // of the label. The value represent the label value for that tag.  
    labels: {[key: string]: string};  
}
```

Response Type

```
interface YamcsInstance {  
  
    // Instance name.  
    name: string;  
    missionDatabase: MissionDatabase;  
    processors: ProcessorInfo[];  
    state: InstanceState;  
  
    // in case the state=FAILED, this field will indicate the cause of the failure  
    // the missionDatabase and other fields may not be filled when this happens  
    failureCause: string;  
    missionTime: string; // RFC 3339 timestamp  
  
    // Labels assigned to this instance. Each entry is keyed by the tag name  
    // of the label. The value represent the label value for that tag.  
    labels: {[key: string]: string};  
  
    // Feature capability hints for client use  
    capabilities: string[];  
  
    // Name of the template, if this instance was generated  
    template: string;  
  
    // Arguments used during template processing, if this instance  
    // was generated  
    templateArgs: {[key: string]: string};  
}
```

(continues on next page)

```

// Whether the template is stil available
templateAvailable: boolean;

// Whether the template has changed since this instance was
// generated
templateChanged: boolean;
}

```

Related Types

```

interface MissionDatabase {

    // This is the config section in mdb.yaml
    configName: string;

    // Root space-system name
    name: string;

    // Root space-system header version
    version: string;
    spaceSystem: SpaceSystemInfo[];
    parameterCount: number;
    containerCount: number;
    commandCount: number;
    algorithmCount: number;
    parameterTypeCount: number;
}

interface SpaceSystemInfo {
    name: string;
    qualifiedName: string;
    shortDescription: string;
    longDescription: string;
    alias: NamedObjectId[];
    version: string;
    history: HistoryInfo[];
    sub: SpaceSystemInfo[];
    ancillaryData: {[key: string]: AncillaryDataInfo};
}

// Used by external clients to identify an item in the Mission Database
// If namespace is set, then the name is that of an alias, rather than
// the qualified name.
interface NamedObjectId {
    name: string;
    namespace: string;
}

interface HistoryInfo {
    version: string;
    date: string;
    message: string;
    author: string;
}

interface ProcessorInfo {

    // Yamcs instance name.
    instance: string;

    // Processor name.
    name: string;
    type: string;
    spec: string;
    creator: string;
    hasAlarms: boolean;
    hasCommanding: boolean;
}

```

(continues on next page)

```

state: ServiceState;
replayRequest: ReplayRequest;
replayState: ReplayState;
services: ServiceInfo[];
persistent: boolean;
time: string; // RFC 3339 timestamp
replay: boolean;
checkCommandClearance: boolean;
}

//used to replay (concurrently) TM packets, parameters and events
interface ReplayRequest {

    // **Required.** The time at which the replay should start.
    start: string; // RFC 3339 timestamp

    // The time at which the replay should stop.
    // If unspecified, the replay will keep going as long as there is remaining data.
    stop: string; // RFC 3339 timestamp

    //what should happen at the end of the replay
    endAction: EndAction;

    //how fast the replay should go
    speed: ReplaySpeed;

    // Reverse the direction of the replay
    reverse: boolean;
    parameterRequest: ParameterReplayRequest;

    // By default all Packets, Events, CommandHistory are part of the replay
    // Unless one or more of the below requests are specified.
    packetRequest: PacketReplayRequest;
    eventRequest: EventReplayRequest;
    commandHistoryRequest: CommandHistoryReplayRequest;
    ppRequest: PpReplayRequest;
}

interface ReplaySpeed {
    type: ReplaySpeedType;
    param: number;
}

interface ParameterReplayRequest {
    nameFilter: NamedObjectId[];
    sendRaw: boolean;
    performMonitoring: boolean;
}

interface PacketReplayRequest {

    // No filter, means all packets for which privileges exist, are sent
    nameFilter: NamedObjectId[];
}

interface EventReplayRequest {
}

interface CommandHistoryReplayRequest {

    // No filter, means all command history entries are sent
    nameFilter: NamedObjectId[];
}

//Request to replay parameters - they can be filtered by the parameter group
interface PpReplayRequest {

    // No filter, means all pp groups are sent
    groupNameFilter: string[];

    // exclude the parameters from these groups

```

```

// this takes precedence over the filter above (i.e. if a group is part of both, it will be excluded)
groupNameExclude: string[];
}

interface ServiceInfo {
    instance: string;
    name: string;
    state: ServiceState;
    className: string;
    processor: string;
}

enum ServiceState {
    NEW = "NEW",
    STARTING = "STARTING",
    RUNNING = "RUNNING",
    STOPPING = "STOPPING",
    TERMINATED = "TERMINATED",
    FAILED = "FAILED",
}

enum EndAction {
    LOOP = "LOOP",
    QUIT = "QUIT",
    STOP = "STOP",
}

enum ReplaySpeedType {
    AFAP = "AFAP",
    FIXED_DELAY = "FIXED_DELAY",
    REALTIME = "REALTIME",
    STEP_BY_STEP = "STEP_BY_STEP",
}

enum ReplayState {

    // just at the beginning or when the replay request (start, stop or packet selection) changes
    INITIALIZATION = "INITIALIZATION",
    RUNNING = "RUNNING",

    // The replay has reached the end with the endaction stop
    STOPPED = "STOPPED",

    // The replay stopped due to an error.
    ERROR = "ERROR",
    PAUSED = "PAUSED",

    // The replay is finished and closed
    CLOSED = "CLOSED",
}

enum ServiceState {
    NEW = "NEW",
    STARTING = "STARTING",
    RUNNING = "RUNNING",
    STOPPING = "STOPPING",
    TERMINATED = "TERMINATED",
    FAILED = "FAILED",
}

enum InstanceState {
    OFFLINE = "OFFLINE",
    INITIALIZING = "INITIALIZING",
    INITIALIZED = "INITIALIZED",
    STARTING = "STARTING",
    RUNNING = "RUNNING",
    STOPPING = "STOPPING",
    FAILED = "FAILED",
}

```

15.9 Reconfigure Instance

Reconfigure a templated instance

Regenerates the instance configuration based on the latest template source, and with optionally modified template variables.

URI Template

```
POST /api/instances/{instance}:reconfigure
```

{instance}

Yamcs instance name.

Request Body

```
interface ReconfigureInstanceRequest {  
  
    // Arguments for substitution in the template definition. Each entry is  
    // keyed by the argument name. The value must be a string.  
    templateArgs: {[key: string]: string};  
  
    // Labels assigned to this instance. Each entry is keyed by the tag name  
    // of the label. The value represent the label value for that tag.  
    labels: {[key: string]: string};  
}
```

Response Type

```
interface YamcsInstance {  
  
    // Instance name.  
    name: string;  
    missionDatabase: MissionDatabase;  
    processors: ProcessorInfo[];  
    state: InstanceState;  
  
    //in case the state=FAILED, this field will indicate the cause of the failure  
    // the missionDatabase and other fields may not be filled when this happens  
    failureCause: string;  
    missionTime: string; // RFC 3339 timestamp  
  
    // Labels assigned to this instance. Each entry is keyed by the tag name  
    // of the label. The value represent the label value for that tag.  
    labels: {[key: string]: string};  
  
    // Feature capability hints for client use  
    capabilities: string[];  
  
    // Name of the template, if this instance was generated  
    template: string;  
  
    // Arguments used during template processing, if this instance  
    // was generated  
    templateArgs: {[key: string]: string};  
  
    // Whether the template is stil available  
    templateAvailable: boolean;  
  
    // Whether the template has changed since this instance was  
    // generated  
    templateChanged: boolean;  
}
```

Related Types

```
interface MissionDatabase {

    // This is the config section in mdb.yaml
    configName: string;

    // Root space-system name
    name: string;

    // Root space-system header version
    version: string;
    spaceSystem: SpaceSystemInfo[];
    parameterCount: number;
    containerCount: number;
    commandCount: number;
    algorithmCount: number;
    parameterTypeCount: number;
}

interface SpaceSystemInfo {
    name: string;
    qualifiedName: string;
    shortDescription: string;
    longDescription: string;
    alias: NamedObjectId[];
    version: string;
    history: HistoryInfo[];
    sub: SpaceSystemInfo[];
    ancillaryData: {[key: string]: AncillaryDataInfo};
}

// Used by external clients to identify an item in the Mission Database
// If namespace is set, then the name is that of an alias, rather than
// the qualified name.
interface NamedObjectId {
    name: string;
    namespace: string;
}

interface HistoryInfo {
    version: string;
    date: string;
    message: string;
    author: string;
}

interface ProcessorInfo {

    // Yamcs instance name.
    instance: string;

    // Processor name.
    name: string;
    type: string;
    spec: string;
    creator: string;
    hasAlarms: boolean;
    hasCommanding: boolean;
    state: ServiceState;
    replayRequest: ReplayRequest;
    replayState: ReplayState;
    services: ServiceInfo[];
    persistent: boolean;
    time: string; // RFC 3339 timestamp
    replay: boolean;
    checkCommandClearance: boolean;
}

//used to replay (concurrently) TM packets, parameters and events
interface ReplayRequest {
```

(continues on next page)


```

// **Required.** The time at which the replay should start.
start: string; // RFC 3339 timestamp

// The time at which the replay should stop.
// If unspecified, the replay will keep going as long as there is remaining data.
stop: string; // RFC 3339 timestamp

//what should happen at the end of the replay
endAction: EndAction;

//how fast the replay should go
speed: ReplaySpeed;

// Reverse the direction of the replay
reverse: boolean;
parameterRequest: ParameterReplayRequest;

// By default all Packets, Events, CommandHistory are part of the replay
// Unless one or more of the below requests are specified.
packetRequest: PacketReplayRequest;
eventRequest: EventReplayRequest;
commandHistoryRequest: CommandHistoryReplayRequest;
ppRequest: PpReplayRequest;
}

interface ReplaySpeed {
    type: ReplaySpeedType;
    param: number;
}

interface ParameterReplayRequest {
    nameFilter: NamedObjectId[];
    sendRaw: boolean;
    performMonitoring: boolean;
}

interface PacketReplayRequest {

    // No filter, means all packets for which privileges exist, are sent
    nameFilter: NamedObjectId[];
}

interface EventReplayRequest {
}

interface CommandHistoryReplayRequest {

    // No filter, means all command history entries are sent
    nameFilter: NamedObjectId[];
}

//Request to replay parameters - they can be filtered by the parameter group
interface PpReplayRequest {

    // No filter, means all pp groups are sent
    groupNameFilter: string[];

    // exclude the parameters from these groups
    // this takes precedence over the filter above (i.e. if a group is part of both, it will be excluded)
    groupNameExclude: string[];
}

interface ServiceInfo {
    instance: string;
    name: string;
    state: ServiceState;
    className: string;
    processor: string;
}

```

```

enum ServiceState {
    NEW = "NEW",
    STARTING = "STARTING",
    RUNNING = "RUNNING",
    STOPPING = "STOPPING",
    TERMINATED = "TERMINATED",
    FAILED = "FAILED",
}

enum EndAction {
    LOOP = "LOOP",
    QUIT = "QUIT",
    STOP = "STOP",
}

enum ReplaySpeedType {
    AFAP = "AFAP",
    FIXED_DELAY = "FIXED_DELAY",
    REALTIME = "REALTIME",
    STEP_BY_STEP = "STEP_BY_STEP",
}

enum ReplayState {

    // just at the beginning or when the replay request (start, stop or packet selection) changes
    INITIALIZATION = "INITIALIZATION",
    RUNNING = "RUNNING",

    // The replay has reached the end with the endaction stop
    STOPPED = "STOPPED",

    // The replay stopped due to an error.
    ERROR = "ERROR",
    PAUSED = "PAUSED",

    // The replay is finished and closed
    CLOSED = "CLOSED",
}

enum ServiceState {
    NEW = "NEW",
    STARTING = "STARTING",
    RUNNING = "RUNNING",
    STOPPING = "STOPPING",
    TERMINATED = "TERMINATED",
    FAILED = "FAILED",
}

enum InstanceState {
    OFFLINE = "OFFLINE",
    INITIALIZING = "INITIALIZING",
    INITIALIZED = "INITIALIZED",
    STARTING = "STARTING",
    RUNNING = "RUNNING",
    STOPPING = "STOPPING",
    FAILED = "FAILED",
}

```

15.10 Start Instance

Start an instance

If the instance is in the RUNNING state, this call will do nothing. Otherwise the instance will be started.

URI Template

POST /api/instances/{instance}:start

{instance}

Yamcs instance name.

Response Type

```
interface YamcsInstance {  
  
    // Instance name.  
    name: string;  
    missionDatabase: MissionDatabase;  
    processors: ProcessorInfo[];  
    state: InstanceState;  
  
    //in case the state=FAILED, this field will indicate the cause of the failure  
    // the missionDatabase and other fields may not be filled when this happens  
    failureCause: string;  
    missionTime: string; // RFC 3339 timestamp  
  
    // Labels assigned to this instance. Each entry is keyed by the tag name  
    // of the label. The value represent the label value for that tag.  
    labels: {[key: string]: string};  
  
    // Feature capability hints for client use  
    capabilities: string[];  
  
    // Name of the template, if this instance was generated  
    template: string;  
  
    // Arguments used during template processing, if this instance  
    // was generated  
    templateArgs: {[key: string]: string};  
  
    // Whether the template is stil available  
    templateAvailable: boolean;  
  
    // Whether the template has changed since this instance was  
    // generated  
    templateChanged: boolean;  
}
```

Related Types

```
interface MissionDatabase {  
  
    // This is the config section in mdb.yaml  
    configName: string;  
  
    // Root space-system name  
    name: string;  
  
    // Root space-system header version  
    version: string;  
    spaceSystem: SpaceSystemInfo[];  
    parameterCount: number;  
    containerCount: number;  
    commandCount: number;  
    algorithmCount: number;  
    parameterTypeCount: number;  
}  
  
interface SpaceSystemInfo {
```

(continues on next page)

```

name: string;
qualifiedName: string;
shortDescription: string;
longDescription: string;
alias: NamedObjectId[];
version: string;
history: HistoryInfo[];
sub: SpaceSystemInfo[];
ancillaryData: {[key: string]: AncillaryDataInfo};
}

// Used by external clients to identify an item in the Mission Database
// If namespace is set, then the name is that of an alias, rather than
// the qualified name.
interface NamedObjectId {
    name: string;
    namespace: string;
}

interface HistoryInfo {
    version: string;
    date: string;
    message: string;
    author: string;
}

interface ProcessorInfo {

    // Yamcs instance name.
    instance: string;

    // Processor name.
    name: string;
    type: string;
    spec: string;
    creator: string;
    hasAlarms: boolean;
    hasCommanding: boolean;
    state: ServiceState;
    replayRequest: ReplayRequest;
    replayState: ReplayState;
    services: ServiceInfo[];
    persistent: boolean;
    time: string; // RFC 3339 timestamp
    replay: boolean;
    checkCommandClearance: boolean;
}

//used to replay (concurrently) TM packets, parameters and events
interface ReplayRequest {

    // **Required.** The time at which the replay should start.
    start: string; // RFC 3339 timestamp

    // The time at which the replay should stop.
    // If unspecified, the replay will keep going as long as there is remaining data.
    stop: string; // RFC 3339 timestamp

    //what should happen at the end of the replay
    endAction: EndAction;

    //how fast the replay should go
    speed: ReplaySpeed;

    // Reverse the direction of the replay
    reverse: boolean;
    parameterRequest: ParameterReplayRequest;

    // By default all Packets, Events, CommandHistory are part of the replay
    // Unless one or more of the below requests are specified.
    packetRequest: PacketReplayRequest;

```

(continued from previous page)

```
eventRequest: EventReplayRequest;  
commandHistoryRequest: CommandHistoryReplayRequest;  
ppRequest: PpReplayRequest;  
}  
  
interface ReplaySpeed {  
    type: ReplaySpeedType;  
    param: number;  
}  
  
interface ParameterReplayRequest {  
    nameFilter: NamedObjectId[];  
    sendRaw: boolean;  
    performMonitoring: boolean;  
}  
  
interface PacketReplayRequest {  
  
    // No filter, means all packets for which privileges exist, are sent  
    nameFilter: NamedObjectId[];  
}  
  
interface EventReplayRequest {  
}  
  
interface CommandHistoryReplayRequest {  
  
    // No filter, means all command history entries are sent  
    nameFilter: NamedObjectId[];  
}  
  
//Request to replay parameters - they can be filtered by the parameter group  
interface PpReplayRequest {  
  
    // No filter, means all pp groups are sent  
    groupNameFilter: string[];  
  
    // exclude the parameters from these groups  
    // this takes precedence over the filter above (i.e. if a group is part of both, it will be excluded)  
    groupNameExclude: string[];  
}  
  
interface ServiceInfo {  
    instance: string;  
    name: string;  
    state: ServiceState;  
    className: string;  
    processor: string;  
}  
  
enum ServiceState {  
    NEW = "NEW",  
    STARTING = "STARTING",  
    RUNNING = "RUNNING",  
    STOPPING = "STOPPING",  
    TERMINATED = "TERMINATED",  
    FAILED = "FAILED",  
}  
  
enum EndAction {  
    LOOP = "LOOP",  
    QUIT = "QUIT",  
    STOP = "STOP",  
}  
  
enum ReplaySpeedType {  
    AFAP = "AFAP",  
    FIXED_DELAY = "FIXED_DELAY",  
    REALTIME = "REALTIME",  
    STEP_BY_STEP = "STEP_BY_STEP",  
}
```

(continues on next page)

(continued from previous page)

```
enum ReplayState {  
  
    // just at the beginning or when the replay request (start, stop or packet selection) changes  
    INITIALIZATION = "INITIALIZATION",  
    RUNNING = "RUNNING",  
  
    // The replay has reached the end with the endaction stop  
    STOPPED = "STOPPED",  
  
    // The replay stopped due to an error.  
    ERROR = "ERROR",  
    PAUSED = "PAUSED",  
  
    // The replay is finished and closed  
    CLOSED = "CLOSED",  
}  
  
enum ServiceState {  
    NEW = "NEW",  
    STARTING = "STARTING",  
    RUNNING = "RUNNING",  
    STOPPING = "STOPPING",  
    TERMINATED = "TERMINATED",  
    FAILED = "FAILED",  
}  
  
enum InstanceState {  
    OFFLINE = "OFFLINE",  
    INITIALIZING = "INITIALIZING",  
    INITIALIZED = "INITIALIZED",  
    STARTING = "STARTING",  
    RUNNING = "RUNNING",  
    STOPPING = "STOPPING",  
    FAILED = "FAILED",  
}
```

15.11 Stop Instance

Stop an instance

Stop all services of the instance. The instance state will be OFFLINE. If the instance state is already OFFLINE, this call will do nothing.

URI Template

```
POST /api/instances/{instance}:stop
```

{instance}

Yamcs instance name.

Response Type

```
interface YamcsInstance {  
  
    // Instance name.  
    name: string;  
    missionDatabase: MissionDatabase;  
    processors: ProcessorInfo[];  
    state: InstanceState;  
}
```

(continues on next page)

(continued from previous page)

```
//in case the state=FAILED, this field will indicate the cause of the failure
// the missionDatabase and other fields may not be filled when this happens
failureCause: string;
missionTime: string; // RFC 3339 timestamp

// Labels assigned to this instance. Each entry is keyed by the tag name
// of the label. The value represent the label value for that tag.
labels: {[key: string]: string};

// Feature capability hints for client use
capabilities: string[];

// Name of the template, if this instance was generated
template: string;

// Arguments used during template processing, if this instance
// was generated
templateArgs: {[key: string]: string};

// Whether the template is stil available
templateAvailable: boolean;

// Whether the template has changed since this instance was
// generated
templateChanged: boolean;
}
```

Related Types

```
interface MissionDatabase {

    // This is the config section in mdb.yaml
    configName: string;

    // Root space-system name
    name: string;

    // Root space-system header version
    version: string;
    spaceSystem: SpaceSystemInfo[];
    parameterCount: number;
    containerCount: number;
    commandCount: number;
    algorithmCount: number;
    parameterTypeCount: number;
}

interface SpaceSystemInfo {
    name: string;
    qualifiedName: string;
    shortDescription: string;
    longDescription: string;
    alias: NamedObjectId[];
    version: string;
    history: HistoryInfo[];
    sub: SpaceSystemInfo[];
    ancillaryData: {[key: string]: AncillaryDataInfo};
}

// Used by external clients to identify an item in the Mission Database
// If namespace is set, then the name is that of an alias, rather than
// the qualified name.
interface NamedObjectId {
    name: string;
    namespace: string;
}

interface HistoryInfo {
```

(continues on next page)

```

version: string;
date: string;
message: string;
author: string;
}

interface ProcessorInfo {

    // Yamcs instance name.
    instance: string;

    // Processor name.
    name: string;
    type: string;
    spec: string;
    creator: string;
    hasAlarms: boolean;
    hasCommanding: boolean;
    state: ServiceState;
    replayRequest: ReplayRequest;
    replayState: ReplayState;
    services: ServiceInfo[];
    persistent: boolean;
    time: string; // RFC 3339 timestamp
    replay: boolean;
    checkCommandClearance: boolean;
}

//used to replay (concurrently) TM packets, parameters and events
interface ReplayRequest {

    // **Required.** The time at which the replay should start.
    start: string; // RFC 3339 timestamp

    // The time at which the replay should stop.
    // If unspecified, the replay will keep going as long as there is remaining data.
    stop: string; // RFC 3339 timestamp

    //what should happen at the end of the replay
    endAction: EndAction;

    //how fast the replay should go
    speed: ReplaySpeed;

    // Reverse the direction of the replay
    reverse: boolean;
    parameterRequest: ParameterReplayRequest;

    // By default all Packets, Events, CommandHistory are part of the replay
    // Unless one or more of the below requests are specified.
    packetRequest: PacketReplayRequest;
    eventRequest: EventReplayRequest;
    commandHistoryRequest: CommandHistoryReplayRequest;
    ppRequest: PpReplayRequest;
}

interface ReplaySpeed {
    type: ReplaySpeedType;
    param: number;
}

interface ParameterReplayRequest {
    nameFilter: NamedObjectId[];
    sendRaw: boolean;
    performMonitoring: boolean;
}

interface PacketReplayRequest {

    // No filter, means all packets for which privileges exist, are sent
    nameFilter: NamedObjectId[];

```



```

}

interface EventReplayRequest {
}

interface CommandHistoryReplayRequest {

    // No filter, means all command history entries are sent
    nameFilter: NamedObjectId[];
}

//Request to replay parameters - they can be filtered by the parameter group
interface PpReplayRequest {

    // No filter, means all pp groups are sent
    groupNameFilter: string[];

    // exclude the parameters from these groups
    // this takes precedence over the filter above (i.e. if a group is part of both, it will be excluded)
    groupNameExclude: string[];
}

interface ServiceInfo {
    instance: string;
    name: string;
    state: ServiceState;
    className: string;
    processor: string;
}

enum ServiceState {
    NEW = "NEW",
    STARTING = "STARTING",
    RUNNING = "RUNNING",
    STOPPING = "STOPPING",
    TERMINATED = "TERMINATED",
    FAILED = "FAILED",
}

enum EndAction {
    LOOP = "LOOP",
    QUIT = "QUIT",
    STOP = "STOP",
}

enum ReplaySpeedType {
    AFAP = "AFAP",
    FIXED_DELAY = "FIXED_DELAY",
    REALTIME = "REALTIME",
    STEP_BY_STEP = "STEP_BY_STEP",
}

enum ReplayState {

    // just at the beginning or when the replay request (start, stop or packet selection) changes
    INITIALIZATION = "INITIALIZATION",
    RUNNING = "RUNNING",

    // The replay has reached the end with the endaction stop
    STOPPED = "STOPPED",

    // The replay stopped due to an error.
    ERROR = "ERROR",
    PAUSED = "PAUSED",

    // The replay is finished and closed
    CLOSED = "CLOSED",
}

enum ServiceState {
    NEW = "NEW",

```

(continues on next page)

(continued from previous page)

```
STARTING = "STARTING",
RUNNING = "RUNNING",
STOPPING = "STOPPING",
TERMINATED = "TERMINATED",
FAILED = "FAILED",
}

enum InstanceState {
    OFFLINE = "OFFLINE",
    INITIALIZING = "INITIALIZING",
    INITIALIZED = "INITIALIZED",
    STARTING = "STARTING",
    RUNNING = "RUNNING",
    STOPPING = "STOPPING",
    FAILED = "FAILED",
}
```

15.12 Restart Instance

Restart an instance

If the instance state is RUNNING, the instance will be stopped and then restarted. Otherwise the instance will be started. Note that the Mission Database will also be reloaded before restart.

URI Template

```
POST /api/instances/{instance}:restart
```

{instance}

Yamcs instance name.

Response Type

```
interface YamcsInstance {

    // Instance name.
    name: string;
    missionDatabase: MissionDatabase;
    processors: ProcessorInfo[];
    state: InstanceState;

    //in case the state=FAILED, this field will indicate the cause of the failure
    // the missionDatabase and other fields may not be filled when this happens
    failureCause: string;
    missionTime: string; // RFC 3339 timestamp

    // Labels assigned to this instance. Each entry is keyed by the tag name
    // of the label. The value represent the label value for that tag.
    labels: {[key: string]: string};

    // Feature capability hints for client use
    capabilities: string[];

    // Name of the template, if this instance was generated
    template: string;

    // Arguments used during template processing, if this instance
    // was generated
    templateArgs: {[key: string]: string};

    // Whether the template is still available
```

(continues on next page)

```

templateAvailable: boolean;

// Whether the template has changed since this instance was
// generated
templateChanged: boolean;
}

```

Related Types

```

interface MissionDatabase {

    // This is the config section in mdb.yaml
    configName: string;

    // Root space-system name
    name: string;

    // Root space-system header version
    version: string;
    spaceSystem: SpaceSystemInfo[];
    parameterCount: number;
    containerCount: number;
    commandCount: number;
    algorithmCount: number;
    parameterTypeCount: number;
}

interface SpaceSystemInfo {
    name: string;
    qualifiedName: string;
    shortDescription: string;
    longDescription: string;
    alias: NamedObjectId[];
    version: string;
    history: HistoryInfo[];
    sub: SpaceSystemInfo[];
    ancillaryData: {[key: string]: AncillaryDataInfo};
}

// Used by external clients to identify an item in the Mission Database
// If namespace is set, then the name is that of an alias, rather than
// the qualified name.
interface NamedObjectId {
    name: string;
    namespace: string;
}

interface HistoryInfo {
    version: string;
    date: string;
    message: string;
    author: string;
}

interface ProcessorInfo {

    // Yamcs instance name.
    instance: string;

    // Processor name.
    name: string;
    type: string;
    spec: string;
    creator: string;
    hasAlarms: boolean;
    hasCommanding: boolean;
    state: ServiceState;
    replayRequest: ReplayRequest;
}

```

(continues on next page)

```

replayState: ReplayState;
services: ServiceInfo[];
persistent: boolean;
time: string; // RFC 3339 timestamp
replay: boolean;
checkCommandClearance: boolean;
}

//used to replay (concurrently) TM packets, parameters and events
interface ReplayRequest {

    // **Required.** The time at which the replay should start.
    start: string; // RFC 3339 timestamp

    // The time at which the replay should stop.
    // If unspecified, the replay will keep going as long as there is remaining data.
    stop: string; // RFC 3339 timestamp

    //what should happen at the end of the replay
    endAction: EndAction;

    //how fast the replay should go
    speed: ReplaySpeed;

    // Reverse the direction of the replay
    reverse: boolean;
    parameterRequest: ParameterReplayRequest;

    // By default all Packets, Events, CommandHistory are part of the replay
    // Unless one or more of the below requests are specified.
    packetRequest: PacketReplayRequest;
    eventRequest: EventReplayRequest;
    commandHistoryRequest: CommandHistoryReplayRequest;
    ppRequest: PpReplayRequest;
}

interface ReplaySpeed {
    type: ReplaySpeedType;
    param: number;
}

interface ParameterReplayRequest {
    nameFilter: NamedObjectId[];
    sendRaw: boolean;
    performMonitoring: boolean;
}

interface PacketReplayRequest {

    // No filter, means all packets for which privileges exist, are sent
    nameFilter: NamedObjectId[];
}

interface EventReplayRequest {
}

interface CommandHistoryReplayRequest {

    // No filter, means all command history entries are sent
    nameFilter: NamedObjectId[];
}

//Request to replay parameters - they can be filtered by the parameter group
interface PpReplayRequest {

    // No filter, means all pp groups are sent
    groupNameFilter: string[];

    // exclude the parameters from these groups
    // this takes precedence over the filter above (i.e. if a group is part of both, it will be excluded)
    groupNameExclude: string[];
}

```

```

}

interface ServiceInfo {
    instance: string;
    name: string;
    state: ServiceState;
    className: string;
    processor: string;
}

enum ServiceState {
    NEW = "NEW",
    STARTING = "STARTING",
    RUNNING = "RUNNING",
    STOPPING = "STOPPING",
    TERMINATED = "TERMINATED",
    FAILED = "FAILED",
}

enum EndAction {
    LOOP = "LOOP",
    QUIT = "QUIT",
    STOP = "STOP",
}

enum ReplaySpeedType {
    AFAP = "AFAP",
    FIXED_DELAY = "FIXED_DELAY",
    REALTIME = "REALTIME",
    STEP_BY_STEP = "STEP_BY_STEP",
}

enum ReplayState {

    // just at the beginning or when the replay request (start, stop or packet selection) changes
    INITIALIZATION = "INITIALIZATION",
    RUNNING = "RUNNING",

    // The replay has reached the end with the endaction stop
    STOPPED = "STOPPED",

    // The replay stopped due to an error.
    ERROR = "ERROR",
    PAUSED = "PAUSED",

    // The replay is finished and closed
    CLOSED = "CLOSED",
}

enum ServiceState {
    NEW = "NEW",
    STARTING = "STARTING",
    RUNNING = "RUNNING",
    STOPPING = "STOPPING",
    TERMINATED = "TERMINATED",
    FAILED = "FAILED",
}

enum InstanceState {
    OFFLINE = "OFFLINE",
    INITIALIZING = "INITIALIZING",
    INITIALIZED = "INITIALIZED",
    STARTING = "STARTING",
    RUNNING = "RUNNING",
    STOPPING = "STOPPING",
    FAILED = "FAILED",
}

```

15.13 List Services

List services

URI Template

```
GET /api/services/{instance}
```

{instance}

Yamcs instance name. Or `_global` for system-wide services.

Response Type

```
interface ListServicesResponse {  
    services: ServiceInfo[];  
}
```

Related Types

```
interface ServiceInfo {  
    instance: string;  
    name: string;  
    state: ServiceState;  
    className: string;  
    processor: string;  
}  
  
enum ServiceState {  
    NEW = "NEW",  
    STARTING = "STARTING",  
    RUNNING = "RUNNING",  
    STOPPING = "STOPPING",  
    TERMINATED = "TERMINATED",  
    FAILED = "FAILED",  
}
```

15.14 Get Service

Get a service

URI Template

```
GET /api/services/{instance}/{name}
```

{instance}

Yamcs instance name. Or `_global` for system-wide services.

{name}

Service name

Response Type

```
interface ServiceInfo {
    instance: string;
    name: string;
    state: ServiceState;
    className: string;
    processor: string;
}
```

Related Types

```
enum ServiceState {
    NEW = "NEW",
    STARTING = "STARTING",
    RUNNING = "RUNNING",
    STOPPING = "STOPPING",
    TERMINATED = "TERMINATED",
    FAILED = "FAILED",
}
```

15.15 Start Service

Start a service

URI Template

```
POST /api/services/{instance}/{name}:start
```

{instance}

Yamcs instance name. Or `_global` for system-wide services.

{name}

Service name

15.16 Stop Service

Stop a service

Once stopped, a service cannot be resumed. Instead a new service instance will be created and started.

URI Template

```
POST /api/services/{instance}/{name}:stop
```

{instance}

Yamcs instance name. Or `_global` for system-wide services.

{name}

Service name

16. Mdb Override

Groups operations that support runtime changes to some parts of the MDB.

These changes are always scoped to a processor, and do not persist across server restarts.

16.1 List Mdb Overrides

List MDB overrides

URI Template

```
GET /api/mdb-overrides/{instance}/{processor}
```

{instance}

Yamcs instance name.

{processor}

Processor name.

Response Type

```
interface ListMdbOverridesResponse {  
  overrides: MdbOverrideInfo[];  
}
```

Related Types

```
interface MdbOverrideInfo {  
  type: OverrideType;  
  algorithmTextOverride: AlgorithmTextOverride;  
}  
  
interface AlgorithmTextOverride {  
  algorithm: string;  
  text: string;  
}  
  
enum OverrideType {  
  ALGORITHM_TEXT = "ALGORITHM_TEXT",  
}
```

16.2 Get Algorithm Overrides

Get overrides for an algorithm

URI Template

```
GET /api/mdb-overrides/{instance}/{processor}/algorithms/{name*}
```

{instance}
Yamcs instance name.

{processor}
Processor name.

{name*}
Algorithm name.

Response Type

```
interface GetAlgorithmOverridesResponse {  
    textOverride: AlgorithmTextOverride;  
}
```

Related Types

```
interface AlgorithmTextOverride {  
    algorithm: string;  
    text: string;  
}
```

16.3 Update Parameter

Update a parameter's definition

URI Template

```
PATCH /api/mdb-overrides/{instance}/{processor}/parameters/{name*}
```

{instance}
Yamcs instance name.

{processor}
Processor name.

{name*}
Alarm name.

Request Body

```
// Used to change calibrators or alarms for one parameter  
interface UpdateParameterRequest {  
  
    // The action by which to modify this alarm.  
    action: ActionType;  
  
    // Used when action = SET_DEFAULT_CALIBRATOR or SET_CALIBRATORS  
    defaultCalibrator: CalibratorInfo;  
  
    // Used when action = SET_CALIBRATORS
```

(continues on next page)

(continued from previous page)

```
contextCalibrator: ContextCalibratorInfo[];

// Used when action = SET_DEFAULT_ALARMS or SET_ALARMS
defaultAlarm: AlarmInfo;

// Used when action = SET_ALARMS
contextAlarm: ContextAlarmInfo[];
}
```

Response Type

```
interface ParameterTypeInfo {
    engType: string;
    dataEncoding: DataEncodingInfo;
    unitSet: UnitInfo[];
    defaultAlarm: AlarmInfo;
    enumValue: EnumValue[];
    absoluteTimeInfo: AbsoluteTimeInfo;
    contextAlarm: ContextAlarmInfo[];
    member: MemberInfo[];
    arrayInfo: ArrayInfo;
    ancillaryData: {[key: string]: AncillaryDataInfo};

    // Provides hints on how to format the engineering
    // value as a string.
    numberFormat: NumberFormatTypeInfo;
}
```

Related Types

```
interface CalibratorInfo {
    polynomialCalibrator: PolynomialCalibratorInfo;
    splineCalibrator: SplineCalibratorInfo;
    javaExpressionCalibrator: JavaExpressionCalibratorInfo;
    type: Type;
}

interface PolynomialCalibratorInfo {
    coefficient: number[];
}

interface SplineCalibratorInfo {
    point: SplinePointInfo[];
}

interface SplinePointInfo {
    raw: number;
    calibrated: number;
}

interface JavaExpressionCalibratorInfo {
    formula: string;
}

interface ContextCalibratorInfo {
    comparison: ComparisonInfo[];
    calibrator: CalibratorInfo;

    // This can be used in UpdateParameterRequest to pass a context
    // that is parsed on the server, according to the rules in the
    // excel spreadsheet. Either this or a comparison has to be
    // used (not both at the same time)
    context: string;
}
```

(continues on next page)

```

interface ComparisonInfo {
    parameter: ParameterInfo;
    operator: OperatorType;
    value: string;
    argument: ArgumentInfo;
}

interface ParameterInfo {
    name: string;
    qualifiedName: string;
    shortDescription: string;
    longDescription: string;
    alias: NamedObjectId[];
    type: ParameterTypeInfo;
    dataSource: DataSourceType;
    usedBy: UsedByInfo;
    ancillaryData: {[key: string]: AncillaryDataInfo};

    // Operations that return aggregate members or array entries
    // may use this field to indicate the path within the parameter.
    path: string[];
}

// Used by external clients to identify an item in the Mission Database
// If namespace is set, then the name is that of an alias, rather than
// the qualified name.
interface NamedObjectId {
    name: string;
    namespace: string;
}

interface UsedByInfo {
    algorithm: AlgorithmInfo[];
    container: ContainerInfo[];
}

interface AlgorithmInfo {
    name: string;
    qualifiedName: string;
    shortDescription: string;
    longDescription: string;
    alias: NamedObjectId[];
    scope: Scope;
    language: string;
    text: string;
    inputParameter: InputParameterInfo[];
    outputParameter: OutputParameterInfo[];
    onParameterUpdate: ParameterInfo[];
    onPeriodicRate: string[]; // String decimal
}

interface InputParameterInfo {
    parameter: ParameterInfo;
    inputName: string;
    parameterInstance: number;
    mandatory: boolean;
    argument: ArgumentInfo;
}

interface ArgumentInfo {
    name: string;
    description: string;

    //optional string type = 3;
    initialValue: string;

    // repeated UnitInfo unitSet = 5;
    type: ArgumentTypeInfo;
}

interface ArgumentTypeInfo {

```

```

engType: string;
dataEncoding: DataEncodingInfo;
unitSet: UnitInfo[];

// Enumeration states (only used by enumerated arguments)
enumValue: EnumValue[];

// Minimum value (only used by integer and float arguments)
rangeMin: number;

// Maximum value (only used by integer and float arguments)
rangeMax: number;

// Member information (only used by aggregate arguments)
member: ArgumentMemberInfo[];

// String representation of a boolean zero (only used by boolean arguments)
zeroStringValue: string;

// String representation of a boolean one (only used by boolean arguments)
oneStringValue: string;

// Minimum character count (only used by string arguments)
minChars: number;

// Maximum character count (only used by string arguments)
maxChars: number;

// True if the engineering type supports signed representation.
// (only used by integer arguments)
signed: boolean;

// Minimum byte count (only used by binary arguments)
minBytes: number;

// Maximum character count (only used by binary arguments)
maxBytes: number;
}

interface DataEncodingInfo {
  type: Type;
  littleEndian: boolean;
  sizeInBits: number;
  encoding: string;
  defaultCalibrator: CalibratorInfo;
  contextCalibrator: ContextCalibratorInfo[];
}

interface UnitInfo {
  unit: string;
}

interface EnumValue {
  value: string; // String decimal
  label: string;
  description: string;
}

interface ArgumentMemberInfo {
  name: string;
  shortDescription: string;
  longDescription: string;
  alias: NamedObjectId[];
  type: ArgumentTypeInfo;
}

interface OutputParameterInfo {
  parameter: ParameterInfo;
  outputName: string;
}

```

```

interface ContainerInfo {
    name: string;
    qualifiedName: string;
    shortDescription: string;
    longDescription: string;
    alias: NamedObjectId[];
    maxInterval: string; // String decimal
    sizeInBits: number;
    baseContainer: ContainerInfo;
    restrictionCriteria: ComparisonInfo[];
    entry: SequenceEntryInfo[];
    usedBy: UsedByInfo;
    ancillaryData: {[key: string]: AncillaryDataInfo};
}

interface SequenceEntryInfo {
    locationInBits: number;
    referenceLocation: ReferenceLocationType;

    // For use in sequence containers
    container: ContainerInfo;
    parameter: ParameterInfo;

    // For use in command containers
    argument: ArgumentInfo;
    fixedValue: FixedValueInfo;
    repeat: RepeatInfo;
}

interface FixedValueInfo {
    name: string;
    hexValue: string;
    sizeInBits: number;
}

interface RepeatInfo {
    fixedCount: string; // String decimal
    dynamicCount: ParameterInfo;
    bitsBetween: number;
}

interface AlarmInfo {
    minViolations: number;
    staticAlarmRange: AlarmRange[];
    enumerationAlarm: EnumerationAlarm[];
}

interface AlarmRange {
    level: AlarmLevelType;
    minInclusive: number;
    maxInclusive: number;
    minExclusive: number;
    maxExclusive: number;
}

interface EnumerationAlarm {
    level: AlarmLevelType;
    label: string;
}

interface ContextAlarmInfo {
    comparison: ComparisonInfo[];
    alarm: AlarmInfo;

    // This can be used in UpdateParameterRequest to pass a context
    // that is parsed on the server, according to the rules in the
    // excel spreadsheet. Either this or a comparison has to be
    // used (not both at the same time)
    context: string;
}

```

```

interface AbsoluteTimeInfo {
    initialValue: string;
    scale: number;
    offset: number;
    offsetFrom: ParameterInfo;
    epoch: string;
}

interface MemberInfo {
    name: string;
    shortDescription: string;
    longDescription: string;
    alias: NamedObjectId[];
    type: ParameterTypeInfo;
}

interface ArrayInfo {
    type: ParameterTypeInfo;
    dimensions: ParameterDimensionInfo[];
}

interface ParameterDimensionInfo {
    fixedValue: string; // String decimal
    parameter: ParameterInfo;
    slope: string; // String decimal
    intercept: string; // String decimal
}

interface NumberFormatTypeInfo {
    numberBase: string;
    minimumFractionDigits: number;
    maximumFractionDigits: number;
    minimumIntegerDigits: number;
    maximumIntegerDigits: number;
    negativeSuffix: string;
    positiveSuffix: string;
    negativePrefix: string;
    positivePrefix: string;
    showThousandsGrouping: boolean;
    notation: string;
}

enum ActionType {

    // Reset all parameter properties (calibrators+alarms) to their default
    // Mission Database value
    RESET = "RESET",

    // Reset calibrators to their default MDB value
    RESET_CALIBRATORS = "RESET_CALIBRATORS",

    // Sets the default calibrator (the contextual ones are unmodified)
    SET_DEFAULT_CALIBRATOR = "SET_DEFAULT_CALIBRATOR",

    // Sets all calibrations (default + contextual), if default is not set,
    // the existing calibration is not modified
    SET_CALIBRATORS = "SET_CALIBRATORS",

    // Reset alarms to their default Mission Database value
    RESET_ALARMS = "RESET_ALARMS",

    // Sets the default alarms (contextual ones are unmodified)
    SET_DEFAULT_ALARMS = "SET_DEFAULT_ALARMS",

    // Sets all alarms (default + contextual), if default is not set, the
    // existing alarm is not modified.
    SET_ALARMS = "SET_ALARMS",
}

enum Type {
    POLYNOMIAL = "POLYNOMIAL",

```

```

    SPLINE = "SPLINE",
    MATH_OPERATION = "MATH_OPERATION",
    JAVA_EXPRESSION = "JAVA_EXPRESSION",
}

enum DataSourceType {
    TELEMETERED = "TELEMETERED",
    DERIVED = "DERIVED",
    CONSTANT = "CONSTANT",
    LOCAL = "LOCAL",
    SYSTEM = "SYSTEM",
    COMMAND = "COMMAND",
    COMMAND_HISTORY = "COMMAND_HISTORY",
    EXTERNAL1 = "EXTERNAL1",
    EXTERNAL2 = "EXTERNAL2",
    EXTERNAL3 = "EXTERNAL3",
}

enum Scope {
    GLOBAL = "GLOBAL",
    COMMAND_VERIFICATION = "COMMAND_VERIFICATION",
    CONTAINER_PROCESSING = "CONTAINER_PROCESSING",
}

enum Type {
    BINARY = "BINARY",
    BOOLEAN = "BOOLEAN",
    FLOAT = "FLOAT",
    INTEGER = "INTEGER",
    STRING = "STRING",
}

enum ReferenceLocationType {
    CONTAINER_START = "CONTAINER_START",
    PREVIOUS_ENTRY = "PREVIOUS_ENTRY",
}

enum OperatorType {
    EQUAL_TO = "EQUAL_TO",
    NOT_EQUAL_TO = "NOT_EQUAL_TO",
    GREATER_THAN = "GREATER_THAN",
    GREATER_THAN_OR_EQUAL_TO = "GREATER_THAN_OR_EQUAL_TO",
    SMALLER_THAN = "SMALLER_THAN",
    SMALLER_THAN_OR_EQUAL_TO = "SMALLER_THAN_OR_EQUAL_TO",
}

enum AlarmLevelType {
    NORMAL = "NORMAL",
    WATCH = "WATCH",
    WARNING = "WARNING",
    DISTRESS = "DISTRESS",
    CRITICAL = "CRITICAL",
    SEVERE = "SEVERE",
}

enum AlarmLevelType {
    NORMAL = "NORMAL",
    WATCH = "WATCH",
    WARNING = "WARNING",
    DISTRESS = "DISTRESS",
    CRITICAL = "CRITICAL",
    SEVERE = "SEVERE",
}

```

16.4 Update Algorithm

Update an algorithm's definition

URI Template

```
PATCH /api/mdb-overrides/{instance}/{processor}/algorithms/{name*}
```

{instance}
Yamcs instance name.

{processor}
Processor name.

{name*}
Algorithm name.

Request Body

```
interface UpdateAlgorithmRequest {  
  
    // The action by which to modify this algorithm  
    action: ActionType;  
  
    // Used when action = SET  
    algorithm: AlgorithmInfo;  
}
```

Related Types

```
interface AlgorithmInfo {  
    name: string;  
    qualifiedName: string;  
    shortDescription: string;  
    longDescription: string;  
    alias: NamedObjectId[];  
    scope: Scope;  
    language: string;  
    text: string;  
    inputParameter: InputParameterInfo[];  
    outputParameter: OutputParameterInfo[];  
    onParameterUpdate: ParameterInfo[];  
    onPeriodicRate: string[]; // String decimal  
}  
  
// Used by external clients to identify an item in the Mission Database  
// If namespace is set, then the name is that of an alias, rather than  
// the qualified name.  
interface NamedObjectId {  
    name: string;  
    namespace: string;  
}  
  
interface InputParameterInfo {  
    parameter: ParameterInfo;  
    inputName: string;  
    parameterInstance: number;  
    mandatory: boolean;  
    argument: ArgumentInfo;  
}  
  
interface ParameterInfo {  
    name: string;  
    qualifiedName: string;  
    shortDescription: string;  
    longDescription: string;  
    alias: NamedObjectId[];  
    type: ParameterTypeInfo;  
    dataSource: DataSourceType;
```

(continues on next page)


```

usedBy: UsedByInfo;
ancillaryData: {[key: string]: AncillaryDataInfo};

// Operations that return aggregate members or array entries
// may use this field to indicate the path within the parameter.
path: string[];
}

interface ParameterTypeInfo {
  engType: string;
  dataEncoding: DataEncodingInfo;
  unitSet: UnitInfo[];
  defaultAlarm: AlarmInfo;
  enumValue: EnumValue[];
  absoluteTimeInfo: AbsoluteTimeInfo;
  contextAlarm: ContextAlarmInfo[];
  member: MemberInfo[];
  arrayInfo: ArrayInfo;
  ancillaryData: {[key: string]: AncillaryDataInfo};

  // Provides hints on how to format the engineering
  // value as a string.
  numberFormat: NumberFormatTypeInfo;
}

interface DataEncodingInfo {
  type: Type;
  littleEndian: boolean;
  sizeInBits: number;
  encoding: string;
  defaultCalibrator: CalibratorInfo;
  contextCalibrator: ContextCalibratorInfo[];
}

interface CalibratorInfo {
  polynomialCalibrator: PolynomialCalibratorInfo;
  splineCalibrator: SplineCalibratorInfo;
  javaExpressionCalibrator: JavaExpressionCalibratorInfo;
  type: Type;
}

interface PolynomialCalibratorInfo {
  coefficient: number[];
}

interface SplineCalibratorInfo {
  point: SplinePointInfo[];
}

interface SplinePointInfo {
  raw: number;
  calibrated: number;
}

interface JavaExpressionCalibratorInfo {
  formula: string;
}

interface ContextCalibratorInfo {
  comparison: ComparisonInfo[];
  calibrator: CalibratorInfo;

  // This can be used in UpdateParameterRequest to pass a context
  // that is parsed on the server, according to the rules in the
  // excel spreadsheet. Either this or a comparison has to be
  // used (not both at the same time)
  context: string;
}

interface ComparisonInfo {
  parameter: ParameterInfo;

```

```

operator: OperatorType;
value: string;
argument: ArgumentInfo;
}

interface ArgumentInfo {
    name: string;
    description: string;

    //optional string type = 3;
    initialValue: string;

    // repeated UnitInfo unitSet = 5;
    type: ArgumentTypeInfo;
}

interface ArgumentTypeInfo {
    engType: string;
    dataEncoding: DataEncodingInfo;
    unitSet: UnitInfo[];

    // Enumeration states (only used by enumerated arguments)
    enumValue: EnumValue[];

    // Minimum value (only used by integer and float arguments)
    rangeMin: number;

    // Maximum value (only used by integer and float arguments)
    rangeMax: number;

    // Member information (only used by aggregate arguments)
    member: ArgumentMemberInfo[];

    // String representation of a boolean zero (only used by boolean arguments)
    zeroStringValue: string;

    // String representation of a boolean one (only used by boolean arguments)
    oneStringValue: string;

    // Minimum character count (only used by string arguments)
    minChars: number;

    // Maximum character count (only used by string arguments)
    maxChars: number;

    // True if the engineering type supports signed representation.
    // (only used by integer arguments)
    signed: boolean;

    // Minimum byte count (only used by binary arguments)
    minBytes: number;

    // Maximum character count (only used by binary arguments)
    maxBytes: number;
}

interface UnitInfo {
    unit: string;
}

interface EnumValue {
    value: string; // String decimal
    label: string;
    description: string;
}

interface ArgumentMemberInfo {
    name: string;
    shortDescription: string;
    longDescription: string;
    alias: NamedObjectId[];
}

```

```

    type: ArgumentTypeInfo;
}

interface AlarmInfo {
    minViolations: number;
    staticAlarmRange: AlarmRange[];
    enumerationAlarm: EnumerationAlarm[];
}

interface AlarmRange {
    level: AlarmLevelType;
    minInclusive: number;
    maxInclusive: number;
    minExclusive: number;
    maxExclusive: number;
}

interface EnumerationAlarm {
    level: AlarmLevelType;
    label: string;
}

interface AbsoluteTimeInfo {
    initialValue: string;
    scale: number;
    offset: number;
    offsetFrom: ParameterInfo;
    epoch: string;
}

interface ContextAlarmInfo {
    comparison: ComparisonInfo[];
    alarm: AlarmInfo;

    // This can be used in UpdateParameterRequest to pass a context
    // that is parsed on the server, according to the rules in the
    // excel spreadsheet. Either this or a comparison has to be
    // used (not both at the same time)
    context: string;
}

interface MemberInfo {
    name: string;
    shortDescription: string;
    longDescription: string;
    alias: NamedObjectId[];
    type: ParameterTypeInfo;
}

interface ArrayInfo {
    type: ParameterTypeInfo;
    dimensions: ParameterDimensionInfo[];
}

interface ParameterDimensionInfo {
    fixedValue: string; // String decimal
    parameter: ParameterInfo;
    slope: string; // String decimal
    intercept: string; // String decimal
}

interface NumberFormatTypeInfo {
    numberBase: string;
    minimumFractionDigits: number;
    maximumFractionDigits: number;
    minimumIntegerDigits: number;
    maximumIntegerDigits: number;
    negativeSuffix: string;
    positiveSuffix: string;
    negativePrefix: string;
    positivePrefix: string;
}

```

```

    showThousandsGrouping: boolean;
    notation: string;
}

interface UsedByInfo {
    algorithm: AlgorithmInfo[];
    container: ContainerInfo[];
}

interface ContainerInfo {
    name: string;
    qualifiedName: string;
    shortDescription: string;
    longDescription: string;
    alias: NamedObjectId[];
    maxInterval: string; // String decimal
    sizeInBits: number;
    baseContainer: ContainerInfo;
    restrictionCriteria: ComparisonInfo[];
    entry: SequenceEntryInfo[];
    usedBy: UsedByInfo;
    ancillaryData: {[key: string]: AncillaryDataInfo};
}

interface SequenceEntryInfo {
    locationInBits: number;
    referenceLocation: ReferenceLocationType;

    // For use in sequence containers
    container: ContainerInfo;
    parameter: ParameterInfo;

    // For use in command containers
    argument: ArgumentInfo;
    fixedValue: FixedValueInfo;
    repeat: RepeatInfo;
}

interface FixedValueInfo {
    name: string;
    hexValue: string;
    sizeInBits: number;
}

interface RepeatInfo {
    fixedCount: string; // String decimal
    dynamicCount: ParameterInfo;
    bitsBetween: number;
}

interface OutputParameterInfo {
    parameter: ParameterInfo;
    outputName: string;
}

enum ActionType {

    // Restores the original MDB definition
    RESET = "RESET",

    // Sets the algorithm text
    SET = "SET",
}

enum Scope {
    GLOBAL = "GLOBAL",
    COMMAND_VERIFICATION = "COMMAND_VERIFICATION",
    CONTAINER_PROCESSING = "CONTAINER_PROCESSING",
}

enum Type {

```

```

    BINARY = "BINARY",
    BOOLEAN = "BOOLEAN",
    FLOAT = "FLOAT",
    INTEGER = "INTEGER",
    STRING = "STRING",
}

enum Type {
    POLYNOMIAL = "POLYNOMIAL",
    SPLINE = "SPLINE",
    MATH_OPERATION = "MATH_OPERATION",
    JAVA_EXPRESSION = "JAVA_EXPRESSION",
}

enum OperatorType {
    EQUAL_TO = "EQUAL_TO",
    NOT_EQUAL_TO = "NOT_EQUAL_TO",
    GREATER_THAN = "GREATER_THAN",
    GREATER_THAN_OR_EQUAL_TO = "GREATER_THAN_OR_EQUAL_TO",
    SMALLER_THAN = "SMALLER_THAN",
    SMALLER_THAN_OR_EQUAL_TO = "SMALLER_THAN_OR_EQUAL_TO",
}

enum AlarmLevelType {
    NORMAL = "NORMAL",
    WATCH = "WATCH",
    WARNING = "WARNING",
    DISTRESS = "DISTRESS",
    CRITICAL = "CRITICAL",
    SEVERE = "SEVERE",
}

enum AlarmLevelType {
    NORMAL = "NORMAL",
    WATCH = "WATCH",
    WARNING = "WARNING",
    DISTRESS = "DISTRESS",
    CRITICAL = "CRITICAL",
    SEVERE = "SEVERE",
}

enum DataSourceType {
    TELEMETERED = "TELEMETERED",
    DERIVED = "DERIVED",
    CONSTANT = "CONSTANT",
    LOCAL = "LOCAL",
    SYSTEM = "SYSTEM",
    COMMAND = "COMMAND",
    COMMAND_HISTORY = "COMMAND_HISTORY",
    EXTERNAL1 = "EXTERNAL1",
    EXTERNAL2 = "EXTERNAL2",
    EXTERNAL3 = "EXTERNAL3",
}

enum ReferenceLocationType {
    CONTAINER_START = "CONTAINER_START",
    PREVIOUS_ENTRY = "PREVIOUS_ENTRY",
}

```

17. Mdb

17.1 Get Mission Database

Get a mission database

URI Template

```
GET /api/mdb/{instance}
```

{instance}

Yamcs instance name.

Response Type

```
interface MissionDatabase {  
  
    // This is the config section in mdb.yaml  
    configName: string;  
  
    // Root space-system name  
    name: string;  
  
    // Root space-system header version  
    version: string;  
    spaceSystem: SpaceSystemInfo[];  
    parameterCount: number;  
    containerCount: number;  
    commandCount: number;  
    algorithmCount: number;  
    parameterTypeCount: number;  
}
```

Related Types

```
interface SpaceSystemInfo {  
    name: string;  
    qualifiedName: string;  
    shortDescription: string;  
    longDescription: string;  
    alias: NamedObjectId[];  
    version: string;  
    history: HistoryInfo[];  
    sub: SpaceSystemInfo[];  
    ancillaryData: {[key: string]: AncillaryDataInfo};  
}  
  
// Used by external clients to identify an item in the Mission Database  
// If namespace is set, then the name is that of an alias, rather than
```

(continues on next page)

(continued from previous page)

```
// the qualified name.
interface NamedObjectId {
    name: string;
    namespace: string;
}

interface HistoryInfo {
    version: string;
    date: string;
    message: string;
    author: string;
}
```

17.2 Export Java Mission Database

Export a java serialized dump of the mission database

URI Template

```
GET /api/mdb/{instance}:exportJava
```

{instance}
Yamcs instance name.

17.3 List Space Systems

List space systems

URI Template

```
GET /api/mdb/{instance}/space-systems
```

{instance}
Yamcs instance name.

Query Parameters

q

The search keywords. This supports searching on the namespace or name.

next

Continuation token returned by a previous page response.

pos

The zero-based row number at which to start outputting results. Default: 0

limit

The maximum number of returned containers per page. Choose this value too high and you risk hitting the maximum response size limit enforced by the server. Default: 100

Response Type

```
interface ListSpaceSystemsResponse {
    spaceSystems: SpaceSystemInfo[];

    // Token indicating the response is only partial. More results can then
    // be obtained by performing the same request (including all original
    // query parameters) and setting the `next` parameter to this token.
    continuationToken: string;

    // The total number of results (across all pages)
    totalSize: number;
}
```

Related Types

```
interface SpaceSystemInfo {
    name: string;
    qualifiedName: string;
    shortDescription: string;
    longDescription: string;
    alias: NamedObjectId[];
    version: string;
    history: HistoryInfo[];
    sub: SpaceSystemInfo[];
    ancillaryData: {[key: string]: AncillaryDataInfo};
}

// Used by external clients to identify an item in the Mission Database
// If namespace is set, then the name is that of an alias, rather than
// the qualified name.
interface NamedObjectId {
    name: string;
    namespace: string;
}

interface HistoryInfo {
    version: string;
    date: string;
    message: string;
    author: string;
}
```

17.4 Get Space System

Get a space system

URI Template

```
GET /api/mdb/{instance}/space-systems/{name*}
```

{instance}
Yamcs instance name.

{name*}
Space-system name.

Response Type

```
interface SpaceSystemInfo {
  name: string;
  qualifiedName: string;
  shortDescription: string;
  longDescription: string;
  alias: NamedObjectId[];
  version: string;
  history: HistoryInfo[];
  sub: SpaceSystemInfo[];
  ancillaryData: {[key: string]: AncillaryDataInfo};
}
```

Related Types

```
// Used by external clients to identify an item in the Mission Database
// If namespace is set, then the name is that of an alias, rather than
// the qualified name.
interface NamedObjectId {
  name: string;
  namespace: string;
}

interface HistoryInfo {
  version: string;
  date: string;
  message: string;
  author: string;
}
```

17.5 List Parameters

List parameters

URI Template

```
GET /api/mdb/{instance}/parameters
```

{instance}

Yamcs instance name.

Query Parameters

q

The search keywords. This supports searching on namespace or name.

searchMembers

When used together with q, include also aggregate members (at any depth) in the search.

Note that this method returns only parameters. Members are part of the type definition.

details

Include details on each returned parameter (this includes long descriptions, aliases, and detailed type information). If unset, only summary information is returned.

type

The parameter types to be included in the result. Valid types are boolean, binary, enumeration, float, integer or string. If unspecified, parameters of all types will be included.

source

Include only parameters of the specified source.

system

List only direct child sub-systems or parameters of the specified system. For example when querying the system “/a” against an MDB with parameters “/a/b/c” and “/a/c”, the result returns the sub system “/a/b” and the parameter “/a/c”.

When system and q are used together, matching parameters at any depth are returned, starting from the specified space system.

next

Continuation token returned by a previous page response.

pos

The zero-based row number at which to start outputting results. Default: 0

limit

The maximum number of returned parameters per page. Choose this value too high and you risk hitting the maximum response size limit enforced by the server. Default: 100

Response Type

```
interface ListParametersResponse {
    spaceSystems: string[];
    parameters: ParameterInfo[];

    // Token indicating the response is only partial. More results can then
    // be obtained by performing the same request (including all original
    // query parameters) and setting the `next` parameter to this token.
    continuationToken: string;

    // The total number of results (across all pages)
    totalSize: number;
}
```

Related Types

```
interface ParameterInfo {
    name: string;
    qualifiedName: string;
    shortDescription: string;
    longDescription: string;
    alias: NamedObjectId[];
    type: ParameterTypeInfo;
    dataSource: DataSourceType;
    usedBy: UsedByInfo;
    ancillaryData: {[key: string]: AncillaryDataInfo};

    // Operations that return aggregate members or array entries
    // may use this field to indicate the path within the parameter.
}
```

(continues on next page)

```

    path: string[];
}

// Used by external clients to identify an item in the Mission Database
// If namespace is set, then the name is that of an alias, rather than
// the qualified name.
interface NamedObjectId {
    name: string;
    namespace: string;
}

interface ParameterTypeInfo {
    engType: string;
    dataEncoding: DataEncodingInfo;
    unitSet: UnitInfo[];
    defaultAlarm: AlarmInfo;
    enumValue: EnumValue[];
    absoluteTimeInfo: AbsoluteTimeInfo;
    contextAlarm: ContextAlarmInfo[];
    member: MemberInfo[];
    arrayInfo: ArrayInfo;
    ancillaryData: {[key: string]: AncillaryDataInfo};

    // Provides hints on how to format the engineering
    // value as a string.
    numberFormat: NumberFormatTypeInfo;
}

interface DataEncodingInfo {
    type: Type;
    littleEndian: boolean;
    sizeInBits: number;
    encoding: string;
    defaultCalibrator: CalibratorInfo;
    contextCalibrator: ContextCalibratorInfo[];
}

interface CalibratorInfo {
    polynomialCalibrator: PolynomialCalibratorInfo;
    splineCalibrator: SplineCalibratorInfo;
    javaExpressionCalibrator: JavaExpressionCalibratorInfo;
    type: Type;
}

interface PolynomialCalibratorInfo {
    coefficient: number[];
}

interface SplineCalibratorInfo {
    point: SplinePointInfo[];
}

interface SplinePointInfo {
    raw: number;
    calibrated: number;
}

interface JavaExpressionCalibratorInfo {
    formula: string;
}

interface ContextCalibratorInfo {
    comparison: ComparisonInfo[];
    calibrator: CalibratorInfo;

    // This can be used in UpdateParameterRequest to pass a context
    // that is parsed on the server, according to the rules in the
    // excel spreadsheet. Either this or a comparison has to be
    // used (not both at the same time)
    context: string;
}

```

```

interface ComparisonInfo {
    parameter: ParameterInfo;
    operator: OperatorType;
    value: string;
    argument: ArgumentInfo;
}

interface ArgumentInfo {
    name: string;
    description: string;

    //optional string type = 3;
    initialValue: string;

    // repeated UnitInfo unitSet = 5;
    type: ArgumentTypeInfo;
}

interface ArgumentTypeInfo {
    engType: string;
    dataEncoding: DataEncodingInfo;
    unitSet: UnitInfo[];

    // Enumeration states (only used by enumerated arguments)
    enumValue: EnumValue[];

    // Minimum value (only used by integer and float arguments)
    rangeMin: number;

    // Maximum value (only used by integer and float arguments)
    rangeMax: number;

    // Member information (only used by aggregate arguments)
    member: ArgumentMemberInfo[];

    // String representation of a boolean zero (only used by boolean arguments)
    zeroStringValue: string;

    // String representation of a boolean one (only used by boolean arguments)
    oneStringValue: string;

    // Minimum character count (only used by string arguments)
    minChars: number;

    // Maximum character count (only used by string arguments)
    maxChars: number;

    // True if the engineering type supports signed representation.
    // (only used by integer arguments)
    signed: boolean;

    // Minimum byte count (only used by binary arguments)
    minBytes: number;

    // Maximum character count (only used by binary arguments)
    maxBytes: number;
}

interface UnitInfo {
    unit: string;
}

interface EnumValue {
    value: string; // String decimal
    label: string;
    description: string;
}

interface ArgumentMemberInfo {
    name: string;

```

```

shortDescription: string;
longDescription: string;
alias: NamedObjectId[];
type: ArgumentTypeInfo;
}

interface AlarmInfo {
  minViolations: number;
  staticAlarmRange: AlarmRange[];
  enumerationAlarm: EnumerationAlarm[];
}

interface AlarmRange {
  level: AlarmLevelType;
  minInclusive: number;
  maxInclusive: number;
  minExclusive: number;
  maxExclusive: number;
}

interface EnumerationAlarm {
  level: AlarmLevelType;
  label: string;
}

interface AbsoluteTimeInfo {
  initialValue: string;
  scale: number;
  offset: number;
  offsetFrom: ParameterInfo;
  epoch: string;
}

interface ContextAlarmInfo {
  comparison: ComparisonInfo[];
  alarm: AlarmInfo;

  // This can be used in UpdateParameterRequest to pass a context
  // that is parsed on the server, according to the rules in the
  // excel spreadsheet. Either this or a comparison has to be
  // used (not both at the same time)
  context: string;
}

interface MemberInfo {
  name: string;
  shortDescription: string;
  longDescription: string;
  alias: NamedObjectId[];
  type: ParameterTypeInfo;
}

interface ArrayInfo {
  type: ParameterTypeInfo;
  dimensions: ParameterDimensionInfo[];
}

interface ParameterDimensionInfo {
  fixedValue: string; // String decimal
  parameter: ParameterInfo;
  slope: string; // String decimal
  intercept: string; // String decimal
}

interface NumberFormatTypeInfo {
  numberBase: string;
  minimumFractionDigits: number;
  maximumFractionDigits: number;
  minimumIntegerDigits: number;
  maximumIntegerDigits: number;
  negativeSuffix: string;
}

```

```

    positiveSuffix: string;
    negativePrefix: string;
    positivePrefix: string;
    showThousandsGrouping: boolean;
    notation: string;
}

interface UsedByInfo {
    algorithm: AlgorithmInfo[];
    container: ContainerInfo[];
}

interface AlgorithmInfo {
    name: string;
    qualifiedName: string;
    shortDescription: string;
    longDescription: string;
    alias: NamedObjectId[];
    scope: Scope;
    language: string;
    text: string;
    inputParameter: InputParameterInfo[];
    outputParameter: OutputParameterInfo[];
    onParameterUpdate: ParameterInfo[];
    onPeriodicRate: string[]; // String decimal
}

interface InputParameterInfo {
    parameter: ParameterInfo;
    inputName: string;
    parameterInstance: number;
    mandatory: boolean;
    argument: ArgumentInfo;
}

interface OutputParameterInfo {
    parameter: ParameterInfo;
    outputName: string;
}

interface ContainerInfo {
    name: string;
    qualifiedName: string;
    shortDescription: string;
    longDescription: string;
    alias: NamedObjectId[];
    maxInterval: string; // String decimal
    sizeInBits: number;
    baseContainer: ContainerInfo;
    restrictionCriteria: ComparisonInfo[];
    entry: SequenceEntryInfo[];
    usedBy: UsedByInfo;
    ancillaryData: {[key: string]: AncillaryDataInfo};
}

interface SequenceEntryInfo {
    locationInBits: number;
    referenceLocation: ReferenceLocationType;

    // For use in sequence containers
    container: ContainerInfo;
    parameter: ParameterInfo;

    // For use in command containers
    argument: ArgumentInfo;
    fixedValue: FixedValueInfo;
    repeat: RepeatInfo;
}

interface FixedValueInfo {
    name: string;

```

```

    hexValue: string;
    sizeInBits: number;
}

interface RepeatInfo {
    fixedCount: string; // String decimal
    dynamicCount: ParameterInfo;
    bitsBetween: number;
}

enum DataSourceType {
    TELEMETERED = "TELEMETERED",
    DERIVED = "DERIVED",
    CONSTANT = "CONSTANT",
    LOCAL = "LOCAL",
    SYSTEM = "SYSTEM",
    COMMAND = "COMMAND",
    COMMAND_HISTORY = "COMMAND_HISTORY",
    EXTERNAL1 = "EXTERNAL1",
    EXTERNAL2 = "EXTERNAL2",
    EXTERNAL3 = "EXTERNAL3",
}

enum Type {
    BINARY = "BINARY",
    BOOLEAN = "BOOLEAN",
    FLOAT = "FLOAT",
    INTEGER = "INTEGER",
    STRING = "STRING",
}

enum Type {
    POLYNOMIAL = "POLYNOMIAL",
    SPLINE = "SPLINE",
    MATH_OPERATION = "MATH_OPERATION",
    JAVA_EXPRESSION = "JAVA_EXPRESSION",
}

enum OperatorType {
    EQUAL_TO = "EQUAL_TO",
    NOT_EQUAL_TO = "NOT_EQUAL_TO",
    GREATER_THAN = "GREATER_THAN",
    GREATER_THAN_OR_EQUAL_TO = "GREATER_THAN_OR_EQUAL_TO",
    SMALLER_THAN = "SMALLER_THAN",
    SMALLER_THAN_OR_EQUAL_TO = "SMALLER_THAN_OR_EQUAL_TO",
}

enum AlarmLevelType {
    NORMAL = "NORMAL",
    WATCH = "WATCH",
    WARNING = "WARNING",
    DISTRESS = "DISTRESS",
    CRITICAL = "CRITICAL",
    SEVERE = "SEVERE",
}

enum AlarmLevelType {
    NORMAL = "NORMAL",
    WATCH = "WATCH",
    WARNING = "WARNING",
    DISTRESS = "DISTRESS",
    CRITICAL = "CRITICAL",
    SEVERE = "SEVERE",
}

enum DataSourceType {
    TELEMETERED = "TELEMETERED",
    DERIVED = "DERIVED",
    CONSTANT = "CONSTANT",
    LOCAL = "LOCAL",
    SYSTEM = "SYSTEM",
}

```

(continued from previous page)

```
COMMAND = "COMMAND",
COMMAND_HISTORY = "COMMAND_HISTORY",
EXTERNAL1 = "EXTERNAL1",
EXTERNAL2 = "EXTERNAL2",
EXTERNAL3 = "EXTERNAL3",
}

enum Scope {
    GLOBAL = "GLOBAL",
    COMMAND_VERIFICATION = "COMMAND_VERIFICATION",
    CONTAINER_PROCESSING = "CONTAINER_PROCESSING",
}

enum ReferenceLocationType {
    CONTAINER_START = "CONTAINER_START",
    PREVIOUS_ENTRY = "PREVIOUS_ENTRY",
}
```

17.6 Get Parameter

Get a parameter

URI Template

```
GET /api/mdb/{instance}/parameters/{name*}
```

{instance}

Yamcs instance name.

{name*}

Parameter name.

Response Type

```
interface ParameterInfo {
    name: string;
    qualifiedName: string;
    shortDescription: string;
    longDescription: string;
    alias: NamedObjectId[];
    type: ParameterTypeInfo;
    dataSource: DataSourceType;
    usedBy: UsedByInfo;
    ancillaryData: {[key: string]: AncillaryDataInfo};

    // Operations that return aggregate members or array entries
    // may use this field to indicate the path within the parameter.
    path: string[];
}
```

Related Types

```
// Used by external clients to identify an item in the Mission Database
// If namespace is set, then the name is that of an alias, rather than
// the qualified name.
interface NamedObjectId {
    name: string;
    namespace: string;
```

(continues on next page)


```

}

interface ParameterTypeInfo {
    engType: string;
    dataEncoding: DataEncodingInfo;
    unitSet: UnitInfo[];
    defaultAlarm: AlarmInfo;
    enumValue: EnumValue[];
    absoluteTimeInfo: AbsoluteTimeInfo;
    contextAlarm: ContextAlarmInfo[];
    member: MemberInfo[];
    arrayInfo: ArrayInfo;
    ancillaryData: {[key: string]: AncillaryDataInfo};

    // Provides hints on how to format the engineering
    // value as a string.
    numberFormat: NumberFormatTypeInfo;
}

interface DataEncodingInfo {
    type: Type;
    littleEndian: boolean;
    sizeInBits: number;
    encoding: string;
    defaultCalibrator: CalibratorInfo;
    contextCalibrator: ContextCalibratorInfo[];
}

interface CalibratorInfo {
    polynomialCalibrator: PolynomialCalibratorInfo;
    splineCalibrator: SplineCalibratorInfo;
    javaExpressionCalibrator: JavaExpressionCalibratorInfo;
    type: Type;
}

interface PolynomialCalibratorInfo {
    coefficient: number[];
}

interface SplineCalibratorInfo {
    point: SplinePointInfo[];
}

interface SplinePointInfo {
    raw: number;
    calibrated: number;
}

interface JavaExpressionCalibratorInfo {
    formula: string;
}

interface ContextCalibratorInfo {
    comparison: ComparisonInfo[];
    calibrator: CalibratorInfo;

    // This can be used in UpdateParameterRequest to pass a context
    // that is parsed on the server, according to the rules in the
    // excel spreadsheet. Either this or a comparison has to be
    // used (not both at the same time)
    context: string;
}

interface ComparisonInfo {
    parameter: ParameterInfo;
    operator: OperatorType;
    value: string;
    argument: ArgumentInfo;
}

interface ArgumentInfo {

```

```

name: string;
description: string;

//optional string type = 3;
initialValue: string;

// repeated UnitInfo unitSet = 5;
type: ArgumentTypeInfo;
}

interface ArgumentTypeInfo {
  engType: string;
  dataEncoding: DataEncodingInfo;
  unitSet: UnitInfo[];

  // Enumeration states (only used by enumerated arguments)
  enumValue: EnumValue[];

  // Minimum value (only used by integer and float arguments)
  rangeMin: number;

  // Maximum value (only used by integer and float arguments)
  rangeMax: number;

  // Member information (only used by aggregate arguments)
  member: ArgumentMemberInfo[];

  // String representation of a boolean zero (only used by boolean arguments)
  zeroStringValue: string;

  // String representation of a boolean one (only used by boolean arguments)
  oneStringValue: string;

  // Minimum character count (only used by string arguments)
  minChars: number;

  // Maximum character count (only used by string arguments)
  maxChars: number;

  // True if the engineering type supports signed representation.
  // (only used by integer arguments)
  signed: boolean;

  // Minimum byte count (only used by binary arguments)
  minBytes: number;

  // Maximum character count (only used by binary arguments)
  maxBytes: number;
}

interface UnitInfo {
  unit: string;
}

interface EnumValue {
  value: string; // String decimal
  label: string;
  description: string;
}

interface ArgumentMemberInfo {
  name: string;
  shortDescription: string;
  longDescription: string;
  alias: NamedObjectId[];
  type: ArgumentTypeInfo;
}

interface AlarmInfo {
  minViolations: number;
  staticAlarmRange: AlarmRange[];
}

```

```

    enumerationAlarm: EnumerationAlarm[];
}

interface AlarmRange {
    level: AlarmLevelType;
    minInclusive: number;
    maxInclusive: number;
    minExclusive: number;
    maxExclusive: number;
}

interface EnumerationAlarm {
    level: AlarmLevelType;
    label: string;
}

interface AbsoluteTimeInfo {
    initialValue: string;
    scale: number;
    offset: number;
    offsetFrom: ParameterInfo;
    epoch: string;
}

interface ContextAlarmInfo {
    comparison: ComparisonInfo[];
    alarm: AlarmInfo;

    // This can be used in UpdateParameterRequest to pass a context
    // that is parsed on the server, according to the rules in the
    // excel spreadsheet. Either this or a comparison has to be
    // used (not both at the same time)
    context: string;
}

interface MemberInfo {
    name: string;
    shortDescription: string;
    longDescription: string;
    alias: NamedObjectId[];
    type: ParameterTypeInfo;
}

interface ArrayInfo {
    type: ParameterTypeInfo;
    dimensions: ParameterDimensionInfo[];
}

interface ParameterDimensionInfo {
    fixedValue: string; // String decimal
    parameter: ParameterInfo;
    slope: string; // String decimal
    intercept: string; // String decimal
}

interface NumberFormatTypeInfo {
    numberBase: string;
    minimumFractionDigits: number;
    maximumFractionDigits: number;
    minimumIntegerDigits: number;
    maximumIntegerDigits: number;
    negativeSuffix: string;
    positiveSuffix: string;
    negativePrefix: string;
    positivePrefix: string;
    showThousandsGrouping: boolean;
    notation: string;
}

interface UsedByInfo {
    algorithm: AlgorithmInfo[];
}

```

```

    container: ContainerInfo[];
}

interface AlgorithmInfo {
    name: string;
    qualifiedName: string;
    shortDescription: string;
    longDescription: string;
    alias: NamedObjectId[];
    scope: Scope;
    language: string;
    text: string;
    inputParameter: InputParameterInfo[];
    outputParameter: OutputParameterInfo[];
    onParameterUpdate: ParameterInfo[];
    onPeriodicRate: string[]; // String decimal
}

interface InputParameterInfo {
    parameter: ParameterInfo;
    inputName: string;
    parameterInstance: number;
    mandatory: boolean;
    argument: ArgumentInfo;
}

interface OutputParameterInfo {
    parameter: ParameterInfo;
    outputName: string;
}

interface ContainerInfo {
    name: string;
    qualifiedName: string;
    shortDescription: string;
    longDescription: string;
    alias: NamedObjectId[];
    maxInterval: string; // String decimal
    sizeInBits: number;
    baseContainer: ContainerInfo;
    restrictionCriteria: ComparisonInfo[];
    entry: SequenceEntryInfo[];
    usedBy: UsedByInfo;
    ancillaryData: {[key: string]: AncillaryDataInfo};
}

interface SequenceEntryInfo {
    locationInBits: number;
    referenceLocation: ReferenceLocationType;

    // For use in sequence containers
    container: ContainerInfo;
    parameter: ParameterInfo;

    // For use in command containers
    argument: ArgumentInfo;
    fixedValue: FixedValueInfo;
    repeat: RepeatInfo;
}

interface FixedValueInfo {
    name: string;
    hexValue: string;
    sizeInBits: number;
}

interface RepeatInfo {
    fixedCount: string; // String decimal
    dynamicCount: ParameterInfo;
    bitsBetween: number;
}

```

```

enum Type {
    BINARY = "BINARY",
    BOOLEAN = "BOOLEAN",
    FLOAT = "FLOAT",
    INTEGER = "INTEGER",
    STRING = "STRING",
}

enum Type {
    POLYNOMIAL = "POLYNOMIAL",
    SPLINE = "SPLINE",
    MATH_OPERATION = "MATH_OPERATION",
    JAVA_EXPRESSION = "JAVA_EXPRESSION",
}

enum OperatorType {
    EQUAL_TO = "EQUAL_TO",
    NOT_EQUAL_TO = "NOT_EQUAL_TO",
    GREATER_THAN = "GREATER_THAN",
    GREATER_THAN_OR_EQUAL_TO = "GREATER_THAN_OR_EQUAL_TO",
    SMALLER_THAN = "SMALLER_THAN",
    SMALLER_THAN_OR_EQUAL_TO = "SMALLER_THAN_OR_EQUAL_TO",
}

enum AlarmLevelType {
    NORMAL = "NORMAL",
    WATCH = "WATCH",
    WARNING = "WARNING",
    DISTRESS = "DISTRESS",
    CRITICAL = "CRITICAL",
    SEVERE = "SEVERE",
}

enum AlarmLevelType {
    NORMAL = "NORMAL",
    WATCH = "WATCH",
    WARNING = "WARNING",
    DISTRESS = "DISTRESS",
    CRITICAL = "CRITICAL",
    SEVERE = "SEVERE",
}

enum DataSourceType {
    TELEMETERED = "TELEMETERED",
    DERIVED = "DERIVED",
    CONSTANT = "CONSTANT",
    LOCAL = "LOCAL",
    SYSTEM = "SYSTEM",
    COMMAND = "COMMAND",
    COMMAND_HISTORY = "COMMAND_HISTORY",
    EXTERNAL1 = "EXTERNAL1",
    EXTERNAL2 = "EXTERNAL2",
    EXTERNAL3 = "EXTERNAL3",
}

enum Scope {
    GLOBAL = "GLOBAL",
    COMMAND_VERIFICATION = "COMMAND_VERIFICATION",
    CONTAINER_PROCESSING = "CONTAINER_PROCESSING",
}

enum ReferenceLocationType {
    CONTAINER_START = "CONTAINER_START",
    PREVIOUS_ENTRY = "PREVIOUS_ENTRY",
}

```

17.7 Batch Get Parameters

Batch get of multiple parameters

URI Template

```
POST /api/mdb/{instance}/parameters:batchGet
```

{instance}

Request Body

```
interface BatchGetParametersRequest {  
    id: NamedObjectId[];  
}
```

Response Type

```
interface BatchGetParametersResponse {  
    response: GetParameterResponse[];  
}
```

Related Types

```
// Used by external clients to identify an item in the Mission Database  
// If namespace is set, then the name is that of an alias, rather than  
// the qualified name.  
interface NamedObjectId {  
    name: string;  
    namespace: string;  
}  
  
interface GetParameterResponse {  
    id: NamedObjectId;  
    parameter: ParameterInfo;  
}  
  
interface ParameterInfo {  
    name: string;  
    qualifiedName: string;  
    shortDescription: string;  
    longDescription: string;  
    alias: NamedObjectId[];  
    type: ParameterTypeInfo;  
    dataSource: DataSourceType;  
    usedBy: UsedByInfo;  
    ancillaryData: {[key: string]: AncillaryDataInfo};  
  
    // Operations that return aggregate members or array entries  
    // may use this field to indicate the path within the parameter.  
    path: string[];  
}  
  
interface ParameterTypeInfo {  
    engType: string;  
    dataEncoding: DataEncodingInfo;  
    unitSet: UnitInfo[];  
    defaultAlarm: AlarmInfo;  
    enumValue: EnumValue[];
```

(continues on next page)

```

absoluteTimeInfo: AbsoluteTimeInfo;
contextAlarm: ContextAlarmInfo[];
member: MemberInfo[];
arrayInfo: ArrayInfo;
ancillaryData: {[key: string]: AncillaryDataInfo};

// Provides hints on how to format the engineering
// value as a string.
numberFormat: NumberFormatTypeInfo;
}

interface DataEncodingInfo {
  type: Type;
  littleEndian: boolean;
  sizeInBits: number;
  encoding: string;
  defaultCalibrator: CalibratorInfo;
  contextCalibrator: ContextCalibratorInfo[];
}

interface CalibratorInfo {
  polynomialCalibrator: PolynomialCalibratorInfo;
  splineCalibrator: SplineCalibratorInfo;
  javaExpressionCalibrator: JavaExpressionCalibratorInfo;
  type: Type;
}

interface PolynomialCalibratorInfo {
  coefficient: number[];
}

interface SplineCalibratorInfo {
  point: SplinePointInfo[];
}

interface SplinePointInfo {
  raw: number;
  calibrated: number;
}

interface JavaExpressionCalibratorInfo {
  formula: string;
}

interface ContextCalibratorInfo {
  comparison: ComparisonInfo[];
  calibrator: CalibratorInfo;

  // This can be used in UpdateParameterRequest to pass a context
// that is parsed on the server, according to the rules in the
// excel spreadsheet. Either this or a comparison has to be
// used (not both at the same time)
  context: string;
}

interface ComparisonInfo {
  parameter: ParameterInfo;
  operator: OperatorType;
  value: string;
  argument: ArgumentInfo;
}

interface ArgumentInfo {
  name: string;
  description: string;

  // optional string type = 3;
  initialValue: string;

  // repeated UnitInfo unitSet = 5;
  type: ArgumentTypeInfo;
}

```

```

}

interface ArgumentTypeInfo {
    engType: string;
    dataEncoding: DataEncodingInfo;
    unitSet: UnitInfo[];

    // Enumeration states (only used by enumerated arguments)
    enumValue: EnumValue[];

    // Minimum value (only used by integer and float arguments)
    rangeMin: number;

    // Maximum value (only used by integer and float arguments)
    rangeMax: number;

    // Member information (only used by aggregate arguments)
    member: ArgumentMemberInfo[];

    // String representation of a boolean zero (only used by boolean arguments)
    zeroStringValue: string;

    // String representation of a boolean one (only used by boolean arguments)
    oneStringValue: string;

    // Minimum character count (only used by string arguments)
    minChars: number;

    // Maximum character count (only used by string arguments)
    maxChars: number;

    // True if the engineering type supports signed representation.
    // (only used by integer arguments)
    signed: boolean;

    // Minimum byte count (only used by binary arguments)
    minBytes: number;

    // Maximum character count (only used by binary arguments)
    maxBytes: number;
}

interface UnitInfo {
    unit: string;
}

interface EnumValue {
    value: string; // String decimal
    label: string;
    description: string;
}

interface ArgumentMemberInfo {
    name: string;
    shortDescription: string;
    longDescription: string;
    alias: NamedObjectId[];
    type: ArgumentTypeInfo;
}

interface AlarmInfo {
    minViolations: number;
    staticAlarmRange: AlarmRange[];
    enumerationAlarm: EnumerationAlarm[];
}

interface AlarmRange {
    level: AlarmLevelType;
    minInclusive: number;
    maxInclusive: number;
    minExclusive: number;
}

```



```

    maxExclusive: number;
}

interface EnumerationAlarm {
    level: AlarmLevelType;
    label: string;
}

interface AbsoluteTimeInfo {
    initialValue: string;
    scale: number;
    offset: number;
    offsetFrom: ParameterInfo;
    epoch: string;
}

interface ContextAlarmInfo {
    comparison: ComparisonInfo[];
    alarm: AlarmInfo;

    // This can be used in UpdateParameterRequest to pass a context
    // that is parsed on the server, according to the rules in the
    // excel spreadsheet. Either this or a comparison has to be
    // used (not both at the same time)
    context: string;
}

interface MemberInfo {
    name: string;
    shortDescription: string;
    longDescription: string;
    alias: NamedObjectId[];
    type: ParameterTypeInfo;
}

interface ArrayInfo {
    type: ParameterTypeInfo;
    dimensions: ParameterDimensionInfo[];
}

interface ParameterDimensionInfo {
    fixedValue: string; // String decimal
    parameter: ParameterInfo;
    slope: string; // String decimal
    intercept: string; // String decimal
}

interface NumberFormatTypeInfo {
    numberBase: string;
    minimumFractionDigits: number;
    maximumFractionDigits: number;
    minimumIntegerDigits: number;
    maximumIntegerDigits: number;
    negativeSuffix: string;
    positiveSuffix: string;
    negativePrefix: string;
    positivePrefix: string;
    showThousandsGrouping: boolean;
    notation: string;
}

interface UsedByInfo {
    algorithm: AlgorithmInfo[];
    container: ContainerInfo[];
}

interface AlgorithmInfo {
    name: string;
    qualifiedName: string;
    shortDescription: string;
    longDescription: string;
}

```

```

    alias: NamedObjectId[];
    scope: Scope;
    language: string;
    text: string;
    inputParameter: InputParameterInfo[];
    outputParameter: OutputParameterInfo[];
    onParameterUpdate: ParameterInfo[];
    onPeriodicRate: string[]; // String decimal
}

interface InputParameterInfo {
    parameter: ParameterInfo;
    inputName: string;
    parameterInstance: number;
    mandatory: boolean;
    argument: ArgumentInfo;
}

interface OutputParameterInfo {
    parameter: ParameterInfo;
    outputName: string;
}

interface ContainerInfo {
    name: string;
    qualifiedName: string;
    shortDescription: string;
    longDescription: string;
    alias: NamedObjectId[];
    maxInterval: string; // String decimal
    sizeInBits: number;
    baseContainer: ContainerInfo;
    restrictionCriteria: ComparisonInfo[];
    entry: SequenceEntryInfo[];
    usedBy: UsedByInfo;
    ancillaryData: {[key: string]: AncillaryDataInfo};
}

interface SequenceEntryInfo {
    locationInBits: number;
    referenceLocation: ReferenceLocationType;

    // For use in sequence containers
    container: ContainerInfo;
    parameter: ParameterInfo;

    // For use in command containers
    argument: ArgumentInfo;
    fixedValue: FixedValueInfo;
    repeat: RepeatInfo;
}

interface FixedValueInfo {
    name: string;
    hexValue: string;
    sizeInBits: number;
}

interface RepeatInfo {
    fixedCount: string; // String decimal
    dynamicCount: ParameterInfo;
    bitsBetween: number;
}

enum Type {
    BINARY = "BINARY",
    BOOLEAN = "BOOLEAN",
    FLOAT = "FLOAT",
    INTEGER = "INTEGER",
    STRING = "STRING",
}

```

```

enum Type {
    POLYNOMIAL = "POLYNOMIAL",
    SPLINE = "SPLINE",
    MATH_OPERATION = "MATH_OPERATION",
    JAVA_EXPRESSION = "JAVA_EXPRESSION",
}

enum OperatorType {
    EQUAL_TO = "EQUAL_TO",
    NOT_EQUAL_TO = "NOT_EQUAL_TO",
    GREATER_THAN = "GREATER_THAN",
    GREATER_THAN_OR_EQUAL_TO = "GREATER_THAN_OR_EQUAL_TO",
    SMALLER_THAN = "SMALLER_THAN",
    SMALLER_THAN_OR_EQUAL_TO = "SMALLER_THAN_OR_EQUAL_TO",
}

enum AlarmLevelType {
    NORMAL = "NORMAL",
    WATCH = "WATCH",
    WARNING = "WARNING",
    DISTRESS = "DISTRESS",
    CRITICAL = "CRITICAL",
    SEVERE = "SEVERE",
}

enum AlarmLevelType {
    NORMAL = "NORMAL",
    WATCH = "WATCH",
    WARNING = "WARNING",
    DISTRESS = "DISTRESS",
    CRITICAL = "CRITICAL",
    SEVERE = "SEVERE",
}

enum DataSourceType {
    TELEMETERED = "TELEMETERED",
    DERIVED = "DERIVED",
    CONSTANT = "CONSTANT",
    LOCAL = "LOCAL",
    SYSTEM = "SYSTEM",
    COMMAND = "COMMAND",
    COMMAND_HISTORY = "COMMAND_HISTORY",
    EXTERNAL1 = "EXTERNAL1",
    EXTERNAL2 = "EXTERNAL2",
    EXTERNAL3 = "EXTERNAL3",
}

enum Scope {
    GLOBAL = "GLOBAL",
    COMMAND_VERIFICATION = "COMMAND_VERIFICATION",
    CONTAINER_PROCESSING = "CONTAINER_PROCESSING",
}

enum ReferenceLocationType {
    CONTAINER_START = "CONTAINER_START",
    PREVIOUS_ENTRY = "PREVIOUS_ENTRY",
}

```

17.8 List Containers

List containers

URI Template

```
GET /api/mdb/{instance}/containers
```

{instance}

Yamcs instance name.

Query Parameters

q

The search keywords. This supports searching on the namespace or name.

system

List only direct child sub-systems or containers of the specified system. For example when querying the system “a” against an MDB with containers “/a/b/c” and “/a/c”, the result returns the sub system “/a/b” and the container “/a/c”.

When system and q are used together, matching containers at any depth are returned, starting from the specified space system.

next

Continuation token returned by a previous page response.

pos

The zero-based row number at which to start outputting results. Default: 0

limit

The maximum number of returned containers per page. Choose this value too high and you risk hitting the maximum response size limit enforced by the server. Default: 100

Response Type

```
interface ListContainersResponse {
    spaceSystems: string[];
    containers: ContainerInfo[];

    // Token indicating the response is only partial. More results can then
    // be obtained by performing the same request (including all original
    // query parameters) and setting the `next` parameter to this token.
    continuationToken: string;

    // The total number of results (across all pages)
    totalSize: number;
}
```

Related Types

```
interface ContainerInfo {
    name: string;
    qualifiedName: string;
    shortDescription: string;
    longDescription: string;
    alias: NamedObjectId[];
    maxInterval: string; // String decimal
```

(continues on next page)

```

    sizeInBits: number;
    baseContainer: ContainerInfo;
    restrictionCriteria: ComparisonInfo[];
    entry: SequenceEntryInfo[];
    usedBy: UsedByInfo;
    ancillaryData: {[key: string]: AncillaryDataInfo};
}

// Used by external clients to identify an item in the Mission Database
// If namespace is set, then the name is that of an alias, rather than
// the qualified name.
interface NamedObjectId {
    name: string;
    namespace: string;
}

interface ComparisonInfo {
    parameter: ParameterInfo;
    operator: OperatorType;
    value: string;
    argument: ArgumentInfo;
}

interface ParameterInfo {
    name: string;
    qualifiedName: string;
    shortDescription: string;
    longDescription: string;
    alias: NamedObjectId[];
    type: ParameterTypeInfo;
    dataSource: DataSourceType;
    usedBy: UsedByInfo;
    ancillaryData: {[key: string]: AncillaryDataInfo};

    // Operations that return aggregate members or array entries
    // may use this field to indicate the path within the parameter.
    path: string[];
}

interface ParameterTypeInfo {
    engType: string;
    dataEncoding: DataEncodingInfo;
    unitSet: UnitInfo[];
    defaultAlarm: AlarmInfo;
    enumValue: EnumValue[];
    absoluteTimeInfo: AbsoluteTimeInfo;
    contextAlarm: ContextAlarmInfo[];
    member: MemberInfo[];
    arrayInfo: ArrayInfo;
    ancillaryData: {[key: string]: AncillaryDataInfo};

    // Provides hints on how to format the engineering
    // value as a string.
    numberFormat: NumberFormatTypeInfo;
}

interface DataEncodingInfo {
    type: Type;
    littleEndian: boolean;
    sizeInBits: number;
    encoding: string;
    defaultCalibrator: CalibratorInfo;
    contextCalibrator: ContextCalibratorInfo[];
}

interface CalibratorInfo {
    polynomialCalibrator: PolynomialCalibratorInfo;
    splineCalibrator: SplineCalibratorInfo;
    javaExpressionCalibrator: JavaExpressionCalibratorInfo;
    type: Type;
}

```

```

interface PolynomialCalibratorInfo {
    coefficient: number[];
}

interface SplineCalibratorInfo {
    point: SplinePointInfo[];
}

interface SplinePointInfo {
    raw: number;
    calibrated: number;
}

interface JavaExpressionCalibratorInfo {
    formula: string;
}

interface ContextCalibratorInfo {
    comparison: ComparisonInfo[];
    calibrator: CalibratorInfo;

    // This can be used in UpdateParameterRequest to pass a context
    // that is parsed on the server, according to the rules in the
    // excel spreadsheet. Either this or a comparison has to be
    // used (not both at the same time)
    context: string;
}

interface UnitInfo {
    unit: string;
}

interface AlarmInfo {
    minViolations: number;
    staticAlarmRange: AlarmRange[];
    enumerationAlarm: EnumerationAlarm[];
}

interface AlarmRange {
    level: AlarmLevelType;
    minInclusive: number;
    maxInclusive: number;
    minExclusive: number;
    maxExclusive: number;
}

interface EnumerationAlarm {
    level: AlarmLevelType;
    label: string;
}

interface EnumValue {
    value: string; // String decimal
    label: string;
    description: string;
}

interface AbsoluteTimeInfo {
    initialValue: string;
    scale: number;
    offset: number;
    offsetFrom: ParameterInfo;
    epoch: string;
}

interface ContextAlarmInfo {
    comparison: ComparisonInfo[];
    alarm: AlarmInfo;

    // This can be used in UpdateParameterRequest to pass a context

```

```

// that is parsed on the server, according to the rules in the
// excel spreadsheet. Either this or a comparison has to be
// used (not both at the same time)
context: string;
}

```

```

interface MemberInfo {
    name: string;
    shortDescription: string;
    longDescription: string;
    alias: NamedObjectId[];
    type: ParameterTypeInfo;
}

```

```

interface ArrayInfo {
    type: ParameterTypeInfo;
    dimensions: ParameterDimensionInfo[];
}

```

```

interface ParameterDimensionInfo {
    fixedValue: string; // String decimal
    parameter: ParameterInfo;
    slope: string; // String decimal
    intercept: string; // String decimal
}

```

```

interface NumberFormatTypeInfo {
    numberBase: string;
    minimumFractionDigits: number;
    maximumFractionDigits: number;
    minimumIntegerDigits: number;
    maximumIntegerDigits: number;
    negativeSuffix: string;
    positiveSuffix: string;
    negativePrefix: string;
    positivePrefix: string;
    showThousandsGrouping: boolean;
    notation: string;
}

```

```

interface UsedByInfo {
    algorithm: AlgorithmInfo[];
    container: ContainerInfo[];
}

```

```

interface AlgorithmInfo {
    name: string;
    qualifiedName: string;
    shortDescription: string;
    longDescription: string;
    alias: NamedObjectId[];
    scope: Scope;
    language: string;
    text: string;
    inputParameter: InputParameterInfo[];
    outputParameter: OutputParameterInfo[];
    onParameterUpdate: ParameterInfo[];
    onPeriodicRate: string[]; // String decimal
}

```

```

interface InputParameterInfo {
    parameter: ParameterInfo;
    inputName: string;
    parameterInstance: number;
    mandatory: boolean;
    argument: ArgumentInfo;
}

```

```

interface ArgumentInfo {
    name: string;
    description: string;
}

```

```

//optional string type = 3;
initialValue: string;

// repeated UnitInfo unitSet = 5;
type: ArgumentTypeInfo;
}

interface ArgumentTypeInfo {
  engType: string;
  dataEncoding: DataEncodingInfo;
  unitSet: UnitInfo[];

  // Enumeration states (only used by enumerated arguments)
  enumValue: EnumValue[];

  // Minimum value (only used by integer and float arguments)
  rangeMin: number;

  // Maximum value (only used by integer and float arguments)
  rangeMax: number;

  // Member information (only used by aggregate arguments)
  member: ArgumentMemberInfo[];

  // String representation of a boolean zero (only used by boolean arguments)
  zeroStringValue: string;

  // String representation of a boolean one (only used by boolean arguments)
  oneStringValue: string;

  // Minimum character count (only used by string arguments)
  minChars: number;

  // Maximum character count (only used by string arguments)
  maxChars: number;

  // True if the engineering type supports signed representation.
  // (only used by integer arguments)
  signed: boolean;

  // Minimum byte count (only used by binary arguments)
  minBytes: number;

  // Maximum character count (only used by binary arguments)
  maxBytes: number;
}

interface ArgumentMemberInfo {
  name: string;
  shortDescription: string;
  longDescription: string;
  alias: NamedObjectId[];
  type: ArgumentTypeInfo;
}

interface OutputParameterInfo {
  parameter: ParameterInfo;
  outputName: string;
}

interface SequenceEntryInfo {
  locationInBits: number;
  referenceLocation: ReferenceLocationType;

  // For use in sequence containers
  container: ContainerInfo;
  parameter: ParameterInfo;

  // For use in command containers
  argument: ArgumentInfo;
}

```



```

    fixedValue: FixedValueInfo;
    repeat: RepeatInfo;
}

interface FixedValueInfo {
    name: string;
    hexValue: string;
    sizeInBits: number;
}

interface RepeatInfo {
    fixedCount: string; // String decimal
    dynamicCount: ParameterInfo;
    bitsBetween: number;
}

enum Type {
    BINARY = "BINARY",
    BOOLEAN = "BOOLEAN",
    FLOAT = "FLOAT",
    INTEGER = "INTEGER",
    STRING = "STRING",
}

enum Type {
    POLYNOMIAL = "POLYNOMIAL",
    SPLINE = "SPLINE",
    MATH_OPERATION = "MATH_OPERATION",
    JAVA_EXPRESSION = "JAVA_EXPRESSION",
}

enum AlarmLevelType {
    NORMAL = "NORMAL",
    WATCH = "WATCH",
    WARNING = "WARNING",
    DISTRESS = "DISTRESS",
    CRITICAL = "CRITICAL",
    SEVERE = "SEVERE",
}

enum AlarmLevelType {
    NORMAL = "NORMAL",
    WATCH = "WATCH",
    WARNING = "WARNING",
    DISTRESS = "DISTRESS",
    CRITICAL = "CRITICAL",
    SEVERE = "SEVERE",
}

enum DataSourceType {
    TELEMETERED = "TELEMETERED",
    DERIVED = "DERIVED",
    CONSTANT = "CONSTANT",
    LOCAL = "LOCAL",
    SYSTEM = "SYSTEM",
    COMMAND = "COMMAND",
    COMMAND_HISTORY = "COMMAND_HISTORY",
    EXTERNAL1 = "EXTERNAL1",
    EXTERNAL2 = "EXTERNAL2",
    EXTERNAL3 = "EXTERNAL3",
}

enum Scope {
    GLOBAL = "GLOBAL",
    COMMAND_VERIFICATION = "COMMAND_VERIFICATION",
    CONTAINER_PROCESSING = "CONTAINER_PROCESSING",
}

enum OperatorType {
    EQUAL_TO = "EQUAL_TO",
    NOT_EQUAL_TO = "NOT_EQUAL_TO",
}

```

(continued from previous page)

```
GREATER_THAN = "GREATER_THAN",
GREATER_THAN_OR_EQUAL_TO = "GREATER_THAN_OR_EQUAL_TO",
SMALLER_THAN = "SMALLER_THAN",
SMALLER_THAN_OR_EQUAL_TO = "SMALLER_THAN_OR_EQUAL_TO",
}

enum ReferenceLocationType {
    CONTAINER_START = "CONTAINER_START",
    PREVIOUS_ENTRY = "PREVIOUS_ENTRY",
}
```

17.9 Get Container

Get a container

URI Template

```
GET /api/mdb/{instance}/containers/{name*}
```

{instance}

Yamcs instance name.

{name*}

Container name.

Response Type

```
interface ContainerInfo {
    name: string;
    qualifiedName: string;
    shortDescription: string;
    longDescription: string;
    alias: NamedObjectId[];
    maxInterval: string; // String decimal
    sizeInBits: number;
    baseContainer: ContainerInfo;
    restrictionCriteria: ComparisonInfo[];
    entry: SequenceEntryInfo[];
    usedBy: UsedByInfo;
    ancillaryData: {[key: string]: AncillaryDataInfo};
}
```

Related Types

```
// Used by external clients to identify an item in the Mission Database
// If namespace is set, then the name is that of an alias, rather than
// the qualified name.
interface NamedObjectId {
    name: string;
    namespace: string;
}

interface ComparisonInfo {
    parameter: ParameterInfo;
    operator: OperatorType;
    value: string;
    argument: ArgumentInfo;
}
```

(continues on next page)

```

interface ParameterInfo {
    name: string;
    qualifiedName: string;
    shortDescription: string;
    longDescription: string;
    alias: NamedObjectId[];
    type: ParameterTypeInfo;
    dataSource: DataSourceType;
    usedBy: UsedByInfo;
    ancillaryData: {[key: string]: AncillaryDataInfo};

    // Operations that return aggregate members or array entries
    // may use this field to indicate the path within the parameter.
    path: string[];
}

interface ParameterTypeInfo {
    engType: string;
    dataEncoding: DataEncodingInfo;
    unitSet: UnitInfo[];
    defaultAlarm: AlarmInfo;
    enumValue: EnumValue[];
    absoluteTimeInfo: AbsoluteTimeInfo;
    contextAlarm: ContextAlarmInfo[];
    member: MemberInfo[];
    arrayInfo: ArrayInfo;
    ancillaryData: {[key: string]: AncillaryDataInfo};

    // Provides hints on how to format the engineering
    // value as a string.
    numberFormat: NumberFormatTypeInfo;
}

interface DataEncodingInfo {
    type: Type;
    littleEndian: boolean;
    sizeInBits: number;
    encoding: string;
    defaultCalibrator: CalibratorInfo;
    contextCalibrator: ContextCalibratorInfo[];
}

interface CalibratorInfo {
    polynomialCalibrator: PolynomialCalibratorInfo;
    splineCalibrator: SplineCalibratorInfo;
    javaExpressionCalibrator: JavaExpressionCalibratorInfo;
    type: Type;
}

interface PolynomialCalibratorInfo {
    coefficient: number[];
}

interface SplineCalibratorInfo {
    point: SplinePointInfo[];
}

interface SplinePointInfo {
    raw: number;
    calibrated: number;
}

interface JavaExpressionCalibratorInfo {
    formula: string;
}

interface ContextCalibratorInfo {
    comparison: ComparisonInfo[];
    calibrator: CalibratorInfo;
}

```

```

// This can be used in UpdateParameterRequest to pass a context
// that is parsed on the server, according to the rules in the
// excel spreadsheet. Either this or a comparison has to be
// used (not both at the same time)
context: string;
}

interface UnitInfo {
    unit: string;
}

interface AlarmInfo {
    minViolations: number;
    staticAlarmRange: AlarmRange[];
    enumerationAlarm: EnumerationAlarm[];
}

interface AlarmRange {
    level: AlarmLevelType;
    minInclusive: number;
    maxInclusive: number;
    minExclusive: number;
    maxExclusive: number;
}

interface EnumerationAlarm {
    level: AlarmLevelType;
    label: string;
}

interface EnumValue {
    value: string; // String decimal
    label: string;
    description: string;
}

interface AbsoluteTimeInfo {
    initialValue: string;
    scale: number;
    offset: number;
    offsetFrom: ParameterInfo;
    epoch: string;
}

interface ContextAlarmInfo {
    comparison: ComparisonInfo[];
    alarm: AlarmInfo;

    // This can be used in UpdateParameterRequest to pass a context
    // that is parsed on the server, according to the rules in the
    // excel spreadsheet. Either this or a comparison has to be
    // used (not both at the same time)
    context: string;
}

interface MemberInfo {
    name: string;
    shortDescription: string;
    longDescription: string;
    alias: NamedObjectId[];
    type: ParameterTypeInfo;
}

interface ArrayInfo {
    type: ParameterTypeInfo;
    dimensions: ParameterDimensionInfo[];
}

interface ParameterDimensionInfo {
    fixedValue: string; // String decimal
    parameter: ParameterInfo;
}

```

```

    slope: string; // String decimal
    intercept: string; // String decimal
}

interface NumberFormatTypeInfo {
    numberBase: string;
    minimumFractionDigits: number;
    maximumFractionDigits: number;
    minimumIntegerDigits: number;
    maximumIntegerDigits: number;
    negativeSuffix: string;
    positiveSuffix: string;
    negativePrefix: string;
    positivePrefix: string;
    showThousandsGrouping: boolean;
    notation: string;
}

interface UsedByInfo {
    algorithm: AlgorithmInfo[];
    container: ContainerInfo[];
}

interface AlgorithmInfo {
    name: string;
    qualifiedName: string;
    shortDescription: string;
    longDescription: string;
    alias: NamedObjectId[];
    scope: Scope;
    language: string;
    text: string;
    inputParameter: InputParameterInfo[];
    outputParameter: OutputParameterInfo[];
    onParameterUpdate: ParameterInfo[];
    onPeriodicRate: string[]; // String decimal
}

interface InputParameterInfo {
    parameter: ParameterInfo;
    inputName: string;
    parameterInstance: number;
    mandatory: boolean;
    argument: ArgumentInfo;
}

interface ArgumentInfo {
    name: string;
    description: string;

    //optional string type = 3;
    initialValue: string;

    // repeated UnitInfo unitSet = 5;
    type: ArgumentTypeInfo;
}

interface ArgumentTypeInfo {
    engType: string;
    dataEncoding: DataEncodingInfo;
    unitSet: UnitInfo[];

    // Enumeration states (only used by enumerated arguments)
    enumValue: EnumValue[];

    // Minimum value (only used by integer and float arguments)
    rangeMin: number;

    // Maximum value (only used by integer and float arguments)
    rangeMax: number;
}

```

```

// Member information (only used by aggregate arguments)
member: ArgumentMemberInfo[];

// String representation of a boolean zero (only used by boolean arguments)
zeroStringValue: string;

// String representation of a boolean one (only used by boolean arguments)
oneStringValue: string;

// Minimum character count (only used by string arguments)
minChars: number;

// Maximum character count (only used by string arguments)
maxChars: number;

// True if the engineering type supports signed representation.
// (only used by integer arguments)
signed: boolean;

// Minimum byte count (only used by binary arguments)
minBytes: number;

// Maximum character count (only used by binary arguments)
maxBytes: number;
}

interface ArgumentMemberInfo {
  name: string;
  shortDescription: string;
  longDescription: string;
  alias: NamedObjectId[];
  type: ArgumentTypeInfo;
}

interface OutputParameterInfo {
  parameter: ParameterInfo;
  outputName: string;
}

interface SequenceEntryInfo {
  locationInBits: number;
  referenceLocation: ReferenceLocationType;

  // For use in sequence containers
  container: ContainerInfo;
  parameter: ParameterInfo;

  // For use in command containers
  argument: ArgumentInfo;
  fixedValue: FixedValueInfo;
  repeat: RepeatInfo;
}

interface FixedValueInfo {
  name: string;
  hexValue: string;
  sizeInBits: number;
}

interface RepeatInfo {
  fixedCount: string; // String decimal
  dynamicCount: ParameterInfo;
  bitsBetween: number;
}

enum Type {
  BINARY = "BINARY",
  BOOLEAN = "BOOLEAN",
  FLOAT = "FLOAT",
  INTEGER = "INTEGER",
  STRING = "STRING",
}

```

```

}

enum Type {
    POLYNOMIAL = "POLYNOMIAL",
    SPLINE = "SPLINE",
    MATH_OPERATION = "MATH_OPERATION",
    JAVA_EXPRESSION = "JAVA_EXPRESSION",
}

enum AlarmLevelType {
    NORMAL = "NORMAL",
    WATCH = "WATCH",
    WARNING = "WARNING",
    DISTRESS = "DISTRESS",
    CRITICAL = "CRITICAL",
    SEVERE = "SEVERE",
}

enum AlarmLevelType {
    NORMAL = "NORMAL",
    WATCH = "WATCH",
    WARNING = "WARNING",
    DISTRESS = "DISTRESS",
    CRITICAL = "CRITICAL",
    SEVERE = "SEVERE",
}

enum DataSourceType {
    TELEMETERED = "TELEMETERED",
    DERIVED = "DERIVED",
    CONSTANT = "CONSTANT",
    LOCAL = "LOCAL",
    SYSTEM = "SYSTEM",
    COMMAND = "COMMAND",
    COMMAND_HISTORY = "COMMAND_HISTORY",
    EXTERNAL1 = "EXTERNAL1",
    EXTERNAL2 = "EXTERNAL2",
    EXTERNAL3 = "EXTERNAL3",
}

enum Scope {
    GLOBAL = "GLOBAL",
    COMMAND_VERIFICATION = "COMMAND_VERIFICATION",
    CONTAINER_PROCESSING = "CONTAINER_PROCESSING",
}

enum OperatorType {
    EQUAL_TO = "EQUAL_TO",
    NOT_EQUAL_TO = "NOT_EQUAL_TO",
    GREATER_THAN = "GREATER_THAN",
    GREATER_THAN_OR_EQUAL_TO = "GREATER_THAN_OR_EQUAL_TO",
    SMALLER_THAN = "SMALLER_THAN",
    SMALLER_THAN_OR_EQUAL_TO = "SMALLER_THAN_OR_EQUAL_TO",
}

enum ReferenceLocationType {
    CONTAINER_START = "CONTAINER_START",
    PREVIOUS_ENTRY = "PREVIOUS_ENTRY",
}

```

17.10 List Commands

List commands

URI Template

```
GET /api/mdb/{instance}/commands
```

{instance}

Yamcs instance name.

Query Parameters

q

The search keywords. This supports searching on namespace or name.

system

List only direct child sub-systems or commands of the specified system. For example when querying the system “/a” against an MDB with commands “/a/b/c” and “/a/c”, the result returns the sub system “/a/b” and the command “/a/c”.

When **system** and **q** are used together, matching commands at any depth are returned, starting from the specified space system.

details

next

Continuation token returned by a previous page response.

pos

The zero-based row number at which to start outputting results. Default: 0

limit

The maximum number of returned commands per page. Choose this value too high and you risk hitting the maximum response size limit enforced by the server. Default: 100

noAbstract

Exclude abstract commands

Response Type

```
interface ListCommandsResponse {
    spaceSystems: string[];
    commands: CommandInfo[];

    // Token indicating the response is only partial. More results can then
    // be obtained by performing the same request (including all original
    // query parameters) and setting the `next` parameter to this token.
    continuationToken: string;

    // The total number of results (across all pages)
    totalSize: number;
}
```


Related Types

```
interface CommandInfo {
    name: string;
    qualifiedName: string;
    shortDescription: string;
    longDescription: string;
    alias: NamedObjectId[];
    baseCommand: CommandInfo;
    abstract: boolean;
    argument: ArgumentInfo[];
    argumentAssignment: ArgumentAssignmentInfo[];
    significance: SignificanceInfo;
    constraint: TransmissionConstraintInfo[];
    commandContainer: CommandContainerInfo;
    verifier: VerifierInfo[];
    ancillaryData: {[key: string]: AncillaryDataInfo};
}

// Used by external clients to identify an item in the Mission Database
// If namespace is set, then the name is that of an alias, rather than
// the qualified name.
interface NamedObjectId {
    name: string;
    namespace: string;
}

interface ArgumentInfo {
    name: string;
    description: string;

    //optional string type = 3;
    initialValue: string;

    // repeated UnitInfo unitSet = 5;
    type: ArgumentTypeInfo;
}

interface ArgumentTypeInfo {
    engType: string;
    dataEncoding: DataEncodingInfo;
    unitSet: UnitInfo[];

    // Enumeration states (only used by enumerated arguments)
    enumValue: EnumValue[];

    // Minimum value (only used by integer and float arguments)
    rangeMin: number;

    // Maximum value (only used by integer and float arguments)
    rangeMax: number;

    // Member information (only used by aggregate arguments)
    member: ArgumentMemberInfo[];

    // String representation of a boolean zero (only used by boolean arguments)
    zeroStringValue: string;

    // String representation of a boolean one (only used by boolean arguments)
    oneStringValue: string;

    // Minimum character count (only used by string arguments)
    minChars: number;

    // Maximum character count (only used by string arguments)
    maxChars: number;

    // True if the engineering type supports signed representation.
    // (only used by integer arguments)
    signed: boolean;
}
```

(continues on next page)

```

// Minimum byte count (only used by binary arguments)
minBytes: number;

// Maximum character count (only used by binary arguments)
maxBytes: number;
}

interface DataEncodingInfo {
  type: Type;
  littleEndian: boolean;
  sizeInBits: number;
  encoding: string;
  defaultCalibrator: CalibratorInfo;
  contextCalibrator: ContextCalibratorInfo[];
}

interface CalibratorInfo {
  polynomialCalibrator: PolynomialCalibratorInfo;
  splineCalibrator: SplineCalibratorInfo;
  javaExpressionCalibrator: JavaExpressionCalibratorInfo;
  type: Type;
}

interface PolynomialCalibratorInfo {
  coefficient: number[];
}

interface SplineCalibratorInfo {
  point: SplinePointInfo[];
}

interface SplinePointInfo {
  raw: number;
  calibrated: number;
}

interface JavaExpressionCalibratorInfo {
  formula: string;
}

interface ContextCalibratorInfo {
  comparison: ComparisonInfo[];
  calibrator: CalibratorInfo;

  // This can be used in UpdateParameterRequest to pass a context
  // that is parsed on the server, according to the rules in the
  // excel spreadsheet. Either this or a comparison has to be
  // used (not both at the same time)
  context: string;
}

interface ComparisonInfo {
  parameter: ParameterInfo;
  operator: OperatorType;
  value: string;
  argument: ArgumentInfo;
}

interface ParameterInfo {
  name: string;
  qualifiedName: string;
  shortDescription: string;
  longDescription: string;
  alias: NamedObjectId[];
  type: ParameterTypeInfo;
  dataSource: DataSourceType;
  usedBy: UsedByInfo;
  ancillaryData: {[key: string]: AncillaryDataInfo};

  // Operations that return aggregate members or array entries
  // may use this field to indicate the path within the parameter.

```

```

    path: string[];
}

interface ParameterTypeInfo {
    engType: string;
    dataEncoding: DataEncodingInfo;
    unitSet: UnitInfo[];
    defaultAlarm: AlarmInfo;
    enumValue: EnumValue[];
    absoluteTimeInfo: AbsoluteTimeInfo;
    contextAlarm: ContextAlarmInfo[];
    member: MemberInfo[];
    arrayInfo: ArrayInfo;
    ancillaryData: {[key: string]: AncillaryDataInfo};

    // Provides hints on how to format the engineering
    // value as a string.
    numberFormat: NumberFormatTypeInfo;
}

interface UnitInfo {
    unit: string;
}

interface AlarmInfo {
    minViolations: number;
    staticAlarmRange: AlarmRange[];
    enumerationAlarm: EnumerationAlarm[];
}

interface AlarmRange {
    level: AlarmLevelType;
    minInclusive: number;
    maxInclusive: number;
    minExclusive: number;
    maxExclusive: number;
}

interface EnumerationAlarm {
    level: AlarmLevelType;
    label: string;
}

interface EnumValue {
    value: string; // String decimal
    label: string;
    description: string;
}

interface AbsoluteTimeInfo {
    initialValue: string;
    scale: number;
    offset: number;
    offsetFrom: ParameterInfo;
    epoch: string;
}

interface ContextAlarmInfo {
    comparison: ComparisonInfo[];
    alarm: AlarmInfo;

    // This can be used in UpdateParameterRequest to pass a context
    // that is parsed on the server, according to the rules in the
    // excel spreadsheet. Either this or a comparison has to be
    // used (not both at the same time)
    context: string;
}

interface MemberInfo {
    name: string;
    shortDescription: string;
}

```

```

    longDescription: string;
    alias: NamedObjectId[];
    type: ParameterTypeInfo;
}

interface ArrayInfo {
    type: ParameterTypeInfo;
    dimensions: ParameterDimensionInfo[];
}

interface ParameterDimensionInfo {
    fixedValue: string; // String decimal
    parameter: ParameterInfo;
    slope: string; // String decimal
    intercept: string; // String decimal
}

interface NumberFormatTypeInfo {
    numberBase: string;
    minimumFractionDigits: number;
    maximumFractionDigits: number;
    minimumIntegerDigits: number;
    maximumIntegerDigits: number;
    negativeSuffix: string;
    positiveSuffix: string;
    negativePrefix: string;
    positivePrefix: string;
    showThousandsGrouping: boolean;
    notation: string;
}

interface UsedByInfo {
    algorithm: AlgorithmInfo[];
    container: ContainerInfo[];
}

interface AlgorithmInfo {
    name: string;
    qualifiedName: string;
    shortDescription: string;
    longDescription: string;
    alias: NamedObjectId[];
    scope: Scope;
    language: string;
    text: string;
    inputParameter: InputParameterInfo[];
    outputParameter: OutputParameterInfo[];
    onParameterUpdate: ParameterInfo[];
    onPeriodicRate: string[]; // String decimal
}

interface InputParameterInfo {
    parameter: ParameterInfo;
    inputName: string;
    parameterInstance: number;
    mandatory: boolean;
    argument: ArgumentInfo;
}

interface OutputParameterInfo {
    parameter: ParameterInfo;
    outputName: string;
}

interface ContainerInfo {
    name: string;
    qualifiedName: string;
    shortDescription: string;
    longDescription: string;
    alias: NamedObjectId[];
    maxInterval: string; // String decimal
}

```

```

    sizeInBits: number;
    baseContainer: ContainerInfo;
    restrictionCriteria: ComparisonInfo[];
    entry: SequenceEntryInfo[];
    usedBy: UsedByInfo;
    ancillaryData: {[key: string]: AncillaryDataInfo};
}

```

```

interface SequenceEntryInfo {
    locationInBits: number;
    referenceLocation: ReferenceLocationType;

    // For use in sequence containers
    container: ContainerInfo;
    parameter: ParameterInfo;

    // For use in command containers
    argument: ArgumentInfo;
    fixedValue: FixedValueInfo;
    repeat: RepeatInfo;
}

```

```

interface FixedValueInfo {
    name: string;
    hexValue: string;
    sizeInBits: number;
}

```

```

interface RepeatInfo {
    fixedCount: string; // String decimal
    dynamicCount: ParameterInfo;
    bitsBetween: number;
}

```

```

interface ArgumentMemberInfo {
    name: string;
    shortDescription: string;
    longDescription: string;
    alias: NamedObjectId[];
    type: ArgumentTypeInfo;
}

```

```

interface ArgumentAssignmentInfo {
    name: string;
    value: string;
}

```

```

interface SignificanceInfo {
    consequenceLevel: SignificanceLevelType;
    reasonForWarning: string;
}

```

```

interface TransmissionConstraintInfo {
    expression: string;
    timeout: string; // String decimal
}

```

```

interface CommandContainerInfo {
    name: string;
    qualifiedName: string;
    shortDescription: string;
    longDescription: string;
    alias: NamedObjectId[];
    sizeInBits: number;
    baseContainer: CommandContainerInfo;
    entry: SequenceEntryInfo[];
}

```

```

interface VerifierInfo {
    stage: string;
    container: ContainerInfo;
}

```

```

    algorithm: AlgorithmInfo;
    onSuccess: TerminationActionType;
    onFail: TerminationActionType;
    onTimeout: TerminationActionType;
    checkWindow: CheckWindowInfo;
}

interface CheckWindowInfo {
    timeToStartChecking: string; // String decimal
    timeToStopChecking: string; // String decimal
    relativeTo: string;
}

enum Type {
    BINARY = "BINARY",
    BOOLEAN = "BOOLEAN",
    FLOAT = "FLOAT",
    INTEGER = "INTEGER",
    STRING = "STRING",
}

enum Type {
    POLYNOMIAL = "POLYNOMIAL",
    SPLINE = "SPLINE",
    MATH_OPERATION = "MATH_OPERATION",
    JAVA_EXPRESSION = "JAVA_EXPRESSION",
}

enum AlarmLevelType {
    NORMAL = "NORMAL",
    WATCH = "WATCH",
    WARNING = "WARNING",
    DISTRESS = "DISTRESS",
    CRITICAL = "CRITICAL",
    SEVERE = "SEVERE",
}

enum AlarmLevelType {
    NORMAL = "NORMAL",
    WATCH = "WATCH",
    WARNING = "WARNING",
    DISTRESS = "DISTRESS",
    CRITICAL = "CRITICAL",
    SEVERE = "SEVERE",
}

enum DataSourceType {
    TELEMETERED = "TELEMETERED",
    DERIVED = "DERIVED",
    CONSTANT = "CONSTANT",
    LOCAL = "LOCAL",
    SYSTEM = "SYSTEM",
    COMMAND = "COMMAND",
    COMMAND_HISTORY = "COMMAND_HISTORY",
    EXTERNAL1 = "EXTERNAL1",
    EXTERNAL2 = "EXTERNAL2",
    EXTERNAL3 = "EXTERNAL3",
}

enum Scope {
    GLOBAL = "GLOBAL",
    COMMAND_VERIFICATION = "COMMAND_VERIFICATION",
    CONTAINER_PROCESSING = "CONTAINER_PROCESSING",
}

enum ReferenceLocationType {
    CONTAINER_START = "CONTAINER_START",
    PREVIOUS_ENTRY = "PREVIOUS_ENTRY",
}

enum OperatorType {

```

(continued from previous page)

```
EQUAL_TO = "EQUAL_TO",
NOT_EQUAL_TO = "NOT_EQUAL_TO",
GREATER_THAN = "GREATER_THAN",
GREATER_THAN_OR_EQUAL_TO = "GREATER_THAN_OR_EQUAL_TO",
SMALLER_THAN = "SMALLER_THAN",
SMALLER_THAN_OR_EQUAL_TO = "SMALLER_THAN_OR_EQUAL_TO",
}

enum SignificanceLevelType {
    NONE = "NONE",
    WATCH = "WATCH",
    WARNING = "WARNING",
    DISTRESS = "DISTRESS",
    CRITICAL = "CRITICAL",
    SEVERE = "SEVERE",
}

enum TerminationActionType {
    SUCCESS = "SUCCESS",
    FAIL = "FAIL",
}
```

17.11 Get Command

Get a command

URI Template

```
GET /api/mdb/{instance}/commands/{name*}
```

{instance}

Yamcs instance name.

{name*}

Command name.

Response Type

```
interface CommandInfo {
    name: string;
    qualifiedName: string;
    shortDescription: string;
    longDescription: string;
    alias: NamedObjectId[];
    baseCommand: CommandInfo;
    abstract: boolean;
    argument: ArgumentInfo[];
    argumentAssignment: ArgumentAssignmentInfo[];
    significance: SignificanceInfo;
    constraint: TransmissionConstraintInfo[];
    commandContainer: CommandContainerInfo;
    verifier: VerifierInfo[];
    ancillaryData: {[key: string]: AncillaryDataInfo};
}
```

Related Types

```

// Used by external clients to identify an item in the Mission Database
// If namespace is set, then the name is that of an alias, rather than
// the qualified name.
interface NamedObjectId {
    name: string;
    namespace: string;
}

interface ArgumentInfo {
    name: string;
    description: string;

    //optional string type = 3;
    initialValue: string;

    // repeated UnitInfo unitSet = 5;
    type: ArgumentTypeInfo;
}

interface ArgumentTypeInfo {
    engType: string;
    dataEncoding: DataEncodingInfo;
    unitSet: UnitInfo[];

    // Enumeration states (only used by enumerated arguments)
    enumValue: EnumValue[];

    // Minimum value (only used by integer and float arguments)
    rangeMin: number;

    // Maximum value (only used by integer and float arguments)
    rangeMax: number;

    // Member information (only used by aggregate arguments)
    member: ArgumentMemberInfo[];

    // String representation of a boolean zero (only used by boolean arguments)
    zeroStringValue: string;

    // String representation of a boolean one (only used by boolean arguments)
    oneStringValue: string;

    // Minimum character count (only used by string arguments)
    minChars: number;

    // Maximum character count (only used by string arguments)
    maxChars: number;

    // True if the engineering type supports signed representation.
    // (only used by integer arguments)
    signed: boolean;

    // Minimum byte count (only used by binary arguments)
    minBytes: number;

    // Maximum character count (only used by binary arguments)
    maxBytes: number;
}

interface DataEncodingInfo {
    type: Type;
    littleEndian: boolean;
    sizeInBits: number;
    encoding: string;
    defaultCalibrator: CalibratorInfo;
    contextCalibrator: ContextCalibratorInfo[];
}

interface CalibratorInfo {
    polynomialCalibrator: PolynomialCalibratorInfo;
    splineCalibrator: SplineCalibratorInfo;
    javaExpressionCalibrator: JavaExpressionCalibratorInfo;
}

```

(continues on next page)


```

    type: Type;
}

interface PolynomialCalibratorInfo {
    coefficient: number[];
}

interface SplineCalibratorInfo {
    point: SplinePointInfo[];
}

interface SplinePointInfo {
    raw: number;
    calibrated: number;
}

interface JavaExpressionCalibratorInfo {
    formula: string;
}

interface ContextCalibratorInfo {
    comparison: ComparisonInfo[];
    calibrator: CalibratorInfo;

    // This can be used in UpdateParameterRequest to pass a context
    // that is parsed on the server, according to the rules in the
    // excel spreadsheet. Either this or a comparison has to be
    // used (not both at the same time)
    context: string;
}

interface ComparisonInfo {
    parameter: ParameterInfo;
    operator: OperatorType;
    value: string;
    argument: ArgumentInfo;
}

interface ParameterInfo {
    name: string;
    qualifiedName: string;
    shortDescription: string;
    longDescription: string;
    alias: NamedObjectId[];
    type: ParameterTypeInfo;
    dataSource: DataSourceType;
    usedBy: UsedByInfo;
    ancillaryData: {[key: string]: AncillaryDataInfo};

    // Operations that return aggregate members or array entries
    // may use this field to indicate the path within the parameter.
    path: string[];
}

interface ParameterTypeInfo {
    engType: string;
    dataEncoding: DataEncodingInfo;
    unitSet: UnitInfo[];
    defaultAlarm: AlarmInfo;
    enumValue: EnumValue[];
    absoluteTimeInfo: AbsoluteTimeInfo;
    contextAlarm: ContextAlarmInfo[];
    member: MemberInfo[];
    arrayInfo: ArrayInfo;
    ancillaryData: {[key: string]: AncillaryDataInfo};

    // Provides hints on how to format the engineering
    // value as a string.
    numberFormat: NumberFormatTypeInfo;
}

```

```

interface UnitInfo {
    unit: string;
}

interface AlarmInfo {
    minViolations: number;
    staticAlarmRange: AlarmRange[];
    enumerationAlarm: EnumerationAlarm[];
}

interface AlarmRange {
    level: AlarmLevelType;
    minInclusive: number;
    maxInclusive: number;
    minExclusive: number;
    maxExclusive: number;
}

interface EnumerationAlarm {
    level: AlarmLevelType;
    label: string;
}

interface EnumValue {
    value: string; // String decimal
    label: string;
    description: string;
}

interface AbsoluteTimeInfo {
    initialValue: string;
    scale: number;
    offset: number;
    offsetFrom: ParameterInfo;
    epoch: string;
}

interface ContextAlarmInfo {
    comparison: ComparisonInfo[];
    alarm: AlarmInfo;

    // This can be used in UpdateParameterRequest to pass a context
    // that is parsed on the server, according to the rules in the
    // excel spreadsheet. Either this or a comparison has to be
    // used (not both at the same time)
    context: string;
}

interface MemberInfo {
    name: string;
    shortDescription: string;
    longDescription: string;
    alias: NamedObjectId[];
    type: ParameterTypeInfo;
}

interface ArrayInfo {
    type: ParameterTypeInfo;
    dimensions: ParameterDimensionInfo[];
}

interface ParameterDimensionInfo {
    fixedValue: string; // String decimal
    parameter: ParameterInfo;
    slope: string; // String decimal
    intercept: string; // String decimal
}

interface NumberFormatTypeInfo {
    numberBase: string;
    minimumFractionDigits: number;
}

```

```

maximumFractionDigits: number;
minimumIntegerDigits: number;
maximumIntegerDigits: number;
negativeSuffix: string;
positiveSuffix: string;
negativePrefix: string;
positivePrefix: string;
showThousandsGrouping: boolean;
notation: string;
}

interface UsedByInfo {
    algorithm: AlgorithmInfo[];
    container: ContainerInfo[];
}

interface AlgorithmInfo {
    name: string;
    qualifiedName: string;
    shortDescription: string;
    longDescription: string;
    alias: NamedObjectId[];
    scope: Scope;
    language: string;
    text: string;
    inputParameter: InputParameterInfo[];
    outputParameter: OutputParameterInfo[];
    onParameterUpdate: ParameterInfo[];
    onPeriodicRate: string[]; // String decimal
}

interface InputParameterInfo {
    parameter: ParameterInfo;
    inputName: string;
    parameterInstance: number;
    mandatory: boolean;
    argument: ArgumentInfo;
}

interface OutputParameterInfo {
    parameter: ParameterInfo;
    outputName: string;
}

interface ContainerInfo {
    name: string;
    qualifiedName: string;
    shortDescription: string;
    longDescription: string;
    alias: NamedObjectId[];
    maxInterval: string; // String decimal
    sizeInBits: number;
    baseContainer: ContainerInfo;
    restrictionCriteria: ComparisonInfo[];
    entry: SequenceEntryInfo[];
    usedBy: UsedByInfo;
    ancillaryData: {[key: string]: AncillaryDataInfo};
}

interface SequenceEntryInfo {
    locationInBits: number;
    referenceLocation: ReferenceLocationType;

    // For use in sequence containers
    container: ContainerInfo;
    parameter: ParameterInfo;

    // For use in command containers
    argument: ArgumentInfo;
    fixedValue: FixedValueInfo;
    repeat: RepeatInfo;

```

```

}

interface FixedValueInfo {
    name: string;
    hexValue: string;
    sizeInBits: number;
}

interface RepeatInfo {
    fixedCount: string; // String decimal
    dynamicCount: ParameterInfo;
    bitsBetween: number;
}

interface ArgumentMemberInfo {
    name: string;
    shortDescription: string;
    longDescription: string;
    alias: NamedObjectId[];
    type: ArgumentTypeInfo;
}

interface ArgumentAssignmentInfo {
    name: string;
    value: string;
}

interface SignificanceInfo {
    consequenceLevel: SignificanceLevelType;
    reasonForWarning: string;
}

interface TransmissionConstraintInfo {
    expression: string;
    timeout: string; // String decimal
}

interface CommandContainerInfo {
    name: string;
    qualifiedName: string;
    shortDescription: string;
    longDescription: string;
    alias: NamedObjectId[];
    sizeInBits: number;
    baseContainer: CommandContainerInfo;
    entry: SequenceEntryInfo[];
}

interface VerifierInfo {
    stage: string;
    container: ContainerInfo;
    algorithm: AlgorithmInfo;
    onSuccess: TerminationActionType;
    onFail: TerminationActionType;
    onTimeout: TerminationActionType;
    checkWindow: CheckWindowInfo;
}

interface CheckWindowInfo {
    timeToStartChecking: string; // String decimal
    timeToStopChecking: string; // String decimal
    relativeTo: string;
}

enum Type {
    BINARY = "BINARY",
    BOOLEAN = "BOOLEAN",
    FLOAT = "FLOAT",
    INTEGER = "INTEGER",
    STRING = "STRING",
}

```

```

enum Type {
    POLYNOMIAL = "POLYNOMIAL",
    SPLINE = "SPLINE",
    MATH_OPERATION = "MATH_OPERATION",
    JAVA_EXPRESSION = "JAVA_EXPRESSION",
}

enum AlarmLevelType {
    NORMAL = "NORMAL",
    WATCH = "WATCH",
    WARNING = "WARNING",
    DISTRESS = "DISTRESS",
    CRITICAL = "CRITICAL",
    SEVERE = "SEVERE",
}

enum AlarmLevelType {
    NORMAL = "NORMAL",
    WATCH = "WATCH",
    WARNING = "WARNING",
    DISTRESS = "DISTRESS",
    CRITICAL = "CRITICAL",
    SEVERE = "SEVERE",
}

enum DataSourceType {
    TELEMETERED = "TELEMETERED",
    DERIVED = "DERIVED",
    CONSTANT = "CONSTANT",
    LOCAL = "LOCAL",
    SYSTEM = "SYSTEM",
    COMMAND = "COMMAND",
    COMMAND_HISTORY = "COMMAND_HISTORY",
    EXTERNAL1 = "EXTERNAL1",
    EXTERNAL2 = "EXTERNAL2",
    EXTERNAL3 = "EXTERNAL3",
}

enum Scope {
    GLOBAL = "GLOBAL",
    COMMAND_VERIFICATION = "COMMAND_VERIFICATION",
    CONTAINER_PROCESSING = "CONTAINER_PROCESSING",
}

enum ReferenceLocationType {
    CONTAINER_START = "CONTAINER_START",
    PREVIOUS_ENTRY = "PREVIOUS_ENTRY",
}

enum OperatorType {
    EQUAL_TO = "EQUAL_TO",
    NOT_EQUAL_TO = "NOT_EQUAL_TO",
    GREATER_THAN = "GREATER_THAN",
    GREATER_THAN_OR_EQUAL_TO = "GREATER_THAN_OR_EQUAL_TO",
    SMALLER_THAN = "SMALLER_THAN",
    SMALLER_THAN_OR_EQUAL_TO = "SMALLER_THAN_OR_EQUAL_TO",
}

enum SignificanceLevelType {
    NONE = "NONE",
    WATCH = "WATCH",
    WARNING = "WARNING",
    DISTRESS = "DISTRESS",
    CRITICAL = "CRITICAL",
    SEVERE = "SEVERE",
}

enum TerminationActionType {
    SUCCESS = "SUCCESS",
    FAIL = "FAIL",
}

```

```
}
```

17.12 List Algorithms

List algorithms

URI Template

```
GET /api/mdb/{instance}/algorithms
```

{instance}

Yamcs instance name.

Query Parameters

q

The search keywords. This supports searching on namespace or name.

system

List only direct child sub-systems or algorithms of the specified system. For example when querying the system “/a” against an MDB with algorithms “/a/b/c” and “/a/c”, the result returns the sub system “/a/b” and the algorithm “/a/c”.

When system and q are used together, matching algorithms at any depth are returned, starting from the specified space system.

next

Continuation token returned by a previous page response.

pos

The zero-based row number at which to start outputting results. Default: 0

limit

The maximum number of returned algorithms per page. Choose this value too high and you risk hitting the maximum response size limit enforced by the server. Default: 100

scope

Include only algorithms of the specified scope

Response Type

```
interface ListAlgorithmsResponse {
    spaceSystems: string[];
    algorithms: AlgorithmInfo[];

    // Token indicating the response is only partial. More results can then
    // be obtained by performing the same request (including all original
    // query parameters) and setting the ``next`` parameter to this token.
```

(continues on next page)

```

continuationToken: string;

// The total number of results (across all pages)
totalSize: number;
}

```

Related Types

```

interface AlgorithmInfo {
  name: string;
  qualifiedName: string;
  shortDescription: string;
  longDescription: string;
  alias: NamedObjectId[];
  scope: Scope;
  language: string;
  text: string;
  inputParameter: InputParameterInfo[];
  outputParameter: OutputParameterInfo[];
  onParameterUpdate: ParameterInfo[];
  onPeriodicRate: string[]; // String decimal
}

// Used by external clients to identify an item in the Mission Database
// If namespace is set, then the name is that of an alias, rather than
// the qualified name.
interface NamedObjectId {
  name: string;
  namespace: string;
}

interface InputParameterInfo {
  parameter: ParameterInfo;
  inputName: string;
  parameterInstance: number;
  mandatory: boolean;
  argument: ArgumentInfo;
}

interface ParameterInfo {
  name: string;
  qualifiedName: string;
  shortDescription: string;
  longDescription: string;
  alias: NamedObjectId[];
  type: ParameterTypeInfo;
  dataSource: DataSourceType;
  usedBy: UsedByInfo;
  ancillaryData: {[key: string]: AncillaryDataInfo};

  // Operations that return aggregate members or array entries
  // may use this field to indicate the path within the parameter.
  path: string[];
}

interface ParameterTypeInfo {
  engType: string;
  dataEncoding: DataEncodingInfo;
  unitSet: UnitInfo[];
  defaultAlarm: AlarmInfo;
  enumValue: EnumValue[];
  absoluteTimeInfo: AbsoluteTimeInfo;
  contextAlarm: ContextAlarmInfo[];
  member: MemberInfo[];
  arrayInfo: ArrayInfo;
  ancillaryData: {[key: string]: AncillaryDataInfo};

  // Provides hints on how to format the engineering

```

(continues on next page)

```

    // value as a string.
    numberFormat: NumberFormatTypeInfo;
}

interface DataEncodingInfo {
    type: Type;
    littleEndian: boolean;
    sizeInBits: number;
    encoding: string;
    defaultCalibrator: CalibratorInfo;
    contextCalibrator: ContextCalibratorInfo[];
}

interface CalibratorInfo {
    polynomialCalibrator: PolynomialCalibratorInfo;
    splineCalibrator: SplineCalibratorInfo;
    javaExpressionCalibrator: JavaExpressionCalibratorInfo;
    type: Type;
}

interface PolynomialCalibratorInfo {
    coefficient: number[];
}

interface SplineCalibratorInfo {
    point: SplinePointInfo[];
}

interface SplinePointInfo {
    raw: number;
    calibrated: number;
}

interface JavaExpressionCalibratorInfo {
    formula: string;
}

interface ContextCalibratorInfo {
    comparison: ComparisonInfo[];
    calibrator: CalibratorInfo;

    // This can be used in UpdateParameterRequest to pass a context
    // that is parsed on the server, according to the rules in the
    // excel spreadsheet. Either this or a comparison has to be
    // used (not both at the same time)
    context: string;
}

interface ComparisonInfo {
    parameter: ParameterInfo;
    operator: OperatorType;
    value: string;
    argument: ArgumentInfo;
}

interface ArgumentInfo {
    name: string;
    description: string;

    //optional string type = 3;
    initialValue: string;

    // repeated UnitInfo unitSet = 5;
    type: ArgumentTypeInfo;
}

interface ArgumentTypeInfo {
    engType: string;
    dataEncoding: DataEncodingInfo;
    unitSet: UnitInfo[];
}

```



```

// Enumeration states (only used by enumerated arguments)
enumValue: EnumValue[];

// Minimum value (only used by integer and float arguments)
rangeMin: number;

// Maximum value (only used by integer and float arguments)
rangeMax: number;

// Member information (only used by aggregate arguments)
member: ArgumentMemberInfo[];

// String representation of a boolean zero (only used by boolean arguments)
zeroStringValue: string;

// String representation of a boolean one (only used by boolean arguments)
oneStringValue: string;

// Minimum character count (only used by string arguments)
minChars: number;

// Maximum character count (only used by string arguments)
maxChars: number;

// True if the engineering type supports signed representation.
// (only used by integer arguments)
signed: boolean;

// Minimum byte count (only used by binary arguments)
minBytes: number;

// Maximum character count (only used by binary arguments)
maxBytes: number;
}

interface UnitInfo {
    unit: string;
}

interface EnumValue {
    value: string; // String decimal
    label: string;
    description: string;
}

interface ArgumentMemberInfo {
    name: string;
    shortDescription: string;
    longDescription: string;
    alias: NamedObjectId[];
    type: ArgumentTypeInfo;
}

interface AlarmInfo {
    minViolations: number;
    staticAlarmRange: AlarmRange[];
    enumerationAlarm: EnumerationAlarm[];
}

interface AlarmRange {
    level: AlarmLevelType;
    minInclusive: number;
    maxInclusive: number;
    minExclusive: number;
    maxExclusive: number;
}

interface EnumerationAlarm {
    level: AlarmLevelType;
    label: string;
}

```

```

interface AbsoluteTimeInfo {
    initialValue: string;
    scale: number;
    offset: number;
    offsetFrom: ParameterInfo;
    epoch: string;
}

interface ContextAlarmInfo {
    comparison: ComparisonInfo[];
    alarm: AlarmInfo;

    // This can be used in UpdateParameterRequest to pass a context
    // that is parsed on the server, according to the rules in the
    // excel spreadsheet. Either this or a comparison has to be
    // used (not both at the same time)
    context: string;
}

interface MemberInfo {
    name: string;
    shortDescription: string;
    longDescription: string;
    alias: NamedObjectId[];
    type: ParameterTypeInfo;
}

interface ArrayInfo {
    type: ParameterTypeInfo;
    dimensions: ParameterDimensionInfo[];
}

interface ParameterDimensionInfo {
    fixedValue: string; // String decimal
    parameter: ParameterInfo;
    slope: string; // String decimal
    intercept: string; // String decimal
}

interface NumberFormatTypeInfo {
    numberBase: string;
    minimumFractionDigits: number;
    maximumFractionDigits: number;
    minimumIntegerDigits: number;
    maximumIntegerDigits: number;
    negativeSuffix: string;
    positiveSuffix: string;
    negativePrefix: string;
    positivePrefix: string;
    showThousandsGrouping: boolean;
    notation: string;
}

interface UsedByInfo {
    algorithm: AlgorithmInfo[];
    container: ContainerInfo[];
}

interface ContainerInfo {
    name: string;
    qualifiedName: string;
    shortDescription: string;
    longDescription: string;
    alias: NamedObjectId[];
    maxInterval: string; // String decimal
    sizeInBits: number;
    baseContainer: ContainerInfo;
    restrictionCriteria: ComparisonInfo[];
    entry: SequenceEntryInfo[];
    usedBy: UsedByInfo;
}

```

```

    ancillaryData: {[key: string]: AncillaryDataInfo};
}

interface SequenceEntryInfo {
    locationInBits: number;
    referenceLocation: ReferenceLocationType;

    // For use in sequence containers
    container: ContainerInfo;
    parameter: ParameterInfo;

    // For use in command containers
    argument: ArgumentInfo;
    fixedValue: FixedValueInfo;
    repeat: RepeatInfo;
}

interface FixedValueInfo {
    name: string;
    hexValue: string;
    sizeInBits: number;
}

interface RepeatInfo {
    fixedCount: string; // String decimal
    dynamicCount: ParameterInfo;
    bitsBetween: number;
}

interface OutputParameterInfo {
    parameter: ParameterInfo;
    outputName: string;
}

enum Scope {
    GLOBAL = "GLOBAL",
    COMMAND_VERIFICATION = "COMMAND_VERIFICATION",
    CONTAINER_PROCESSING = "CONTAINER_PROCESSING",
}

enum Scope {
    GLOBAL = "GLOBAL",
    COMMAND_VERIFICATION = "COMMAND_VERIFICATION",
    CONTAINER_PROCESSING = "CONTAINER_PROCESSING",
}

enum Type {
    BINARY = "BINARY",
    BOOLEAN = "BOOLEAN",
    FLOAT = "FLOAT",
    INTEGER = "INTEGER",
    STRING = "STRING",
}

enum Type {
    POLYNOMIAL = "POLYNOMIAL",
    SPLINE = "SPLINE",
    MATH_OPERATION = "MATH_OPERATION",
    JAVA_EXPRESSION = "JAVA_EXPRESSION",
}

enum OperatorType {
    EQUAL_TO = "EQUAL_TO",
    NOT_EQUAL_TO = "NOT_EQUAL_TO",
    GREATER_THAN = "GREATER_THAN",
    GREATER_THAN_OR_EQUAL_TO = "GREATER_THAN_OR_EQUAL_TO",
    SMALLER_THAN = "SMALLER_THAN",
    SMALLER_THAN_OR_EQUAL_TO = "SMALLER_THAN_OR_EQUAL_TO",
}

enum AlarmLevelType {

```

(continued from previous page)

```
NORMAL = "NORMAL",
WATCH = "WATCH",
WARNING = "WARNING",
DISTRESS = "DISTRESS",
CRITICAL = "CRITICAL",
SEVERE = "SEVERE",
}

enum AlarmLevelType {
    NORMAL = "NORMAL",
    WATCH = "WATCH",
    WARNING = "WARNING",
    DISTRESS = "DISTRESS",
    CRITICAL = "CRITICAL",
    SEVERE = "SEVERE",
}

enum DataSourceType {
    TELEMETERED = "TELEMETERED",
    DERIVED = "DERIVED",
    CONSTANT = "CONSTANT",
    LOCAL = "LOCAL",
    SYSTEM = "SYSTEM",
    COMMAND = "COMMAND",
    COMMAND_HISTORY = "COMMAND_HISTORY",
    EXTERNAL1 = "EXTERNAL1",
    EXTERNAL2 = "EXTERNAL2",
    EXTERNAL3 = "EXTERNAL3",
}

enum ReferenceLocationType {
    CONTAINER_START = "CONTAINER_START",
    PREVIOUS_ENTRY = "PREVIOUS_ENTRY",
}
```

17.13 Get Algorithm

Get an algorithm

URI Template

```
GET /api/mdb/{instance}/algorithms/{name*}
```

{instance}

Yamcs instance name.

{name*}

Algorithm name.

Response Type

```
interface AlgorithmInfo {
    name: string;
    qualifiedName: string;
    shortDescription: string;
    longDescription: string;
    alias: NamedObjectId[];
    scope: Scope;
    language: string;
    text: string;
    inputParameter: InputParameterInfo[];
```

(continues on next page)

```

outputParameter: OutputParameterInfo[];
onParameterUpdate: ParameterInfo[];
onPeriodicRate: string[]; // String decimal
}

```

Related Types

```

// Used by external clients to identify an item in the Mission Database
// If namespace is set, then the name is that of an alias, rather than
// the qualified name.
interface NamedObjectId {
    name: string;
    namespace: string;
}

interface InputParameterInfo {
    parameter: ParameterInfo;
    inputName: string;
    parameterInstance: number;
    mandatory: boolean;
    argument: ArgumentInfo;
}

interface ParameterInfo {
    name: string;
    qualifiedName: string;
    shortDescription: string;
    longDescription: string;
    alias: NamedObjectId[];
    type: ParameterTypeInfo;
    dataSource: DataSourceType;
    usedBy: UsedByInfo;
    ancillaryData: {[key: string]: AncillaryDataInfo};

    // Operations that return aggregate members or array entries
    // may use this field to indicate the path within the parameter.
    path: string[];
}

interface ParameterTypeInfo {
    engType: string;
    dataEncoding: DataEncodingInfo;
    unitSet: UnitInfo[];
    defaultAlarm: AlarmInfo;
    enumValue: EnumValue[];
    absoluteTimeInfo: AbsoluteTimeInfo;
    contextAlarm: ContextAlarmInfo[];
    member: MemberInfo[];
    arrayInfo: ArrayInfo;
    ancillaryData: {[key: string]: AncillaryDataInfo};

    // Provides hints on how to format the engineering
    // value as a string.
    numberFormat: NumberFormatTypeInfo;
}

interface DataEncodingInfo {
    type: Type;
    littleEndian: boolean;
    sizeInBits: number;
    encoding: string;
    defaultCalibrator: CalibratorInfo;
    contextCalibrator: ContextCalibratorInfo[];
}

interface CalibratorInfo {
    polynomialCalibrator: PolynomialCalibratorInfo;
    splineCalibrator: SplineCalibratorInfo;
}

```

(continues on next page)

```

javaExpressionCalibrator: JavaExpressionCalibratorInfo;
type: Type;
}

interface PolynomialCalibratorInfo {
    coefficient: number[];
}

interface SplineCalibratorInfo {
    point: SplinePointInfo[];
}

interface SplinePointInfo {
    raw: number;
    calibrated: number;
}

interface JavaExpressionCalibratorInfo {
    formula: string;
}

interface ContextCalibratorInfo {
    comparison: ComparisonInfo[];
    calibrator: CalibratorInfo;

    // This can be used in UpdateParameterRequest to pass a context
    // that is parsed on the server, according to the rules in the
    // excel spreadsheet. Either this or a comparison has to be
    // used (not both at the same time)
    context: string;
}

interface ComparisonInfo {
    parameter: ParameterInfo;
    operator: OperatorType;
    value: string;
    argument: ArgumentInfo;
}

interface ArgumentInfo {
    name: string;
    description: string;

    //optional string type = 3;
    initialValue: string;

    // repeated UnitInfo unitSet = 5;
    type: ArgumentTypeInfo;
}

interface ArgumentTypeInfo {
    engType: string;
    dataEncoding: DataEncodingInfo;
    unitSet: UnitInfo[];

    // Enumeration states (only used by enumerated arguments)
    enumValue: EnumValue[];

    // Minimum value (only used by integer and float arguments)
    rangeMin: number;

    // Maximum value (only used by integer and float arguments)
    rangeMax: number;

    // Member information (only used by aggregate arguments)
    member: ArgumentMemberInfo[];

    // String representation of a boolean zero (only used by boolean arguments)
    zeroStringValue: string;

    // String representation of a boolean one (only used by boolean arguments)

```

```

oneStringValue: string;

// Minimum character count (only used by string arguments)
minChars: number;

// Maximum character count (only used by string arguments)
maxChars: number;

// True if the engineering type supports signed representation.
// (only used by integer arguments)
signed: boolean;

// Minimum byte count (only used by binary arguments)
minBytes: number;

// Maximum character count (only used by binary arguments)
maxBytes: number;
}

interface UnitInfo {
    unit: string;
}

interface EnumValue {
    value: string; // String decimal
    label: string;
    description: string;
}

interface ArgumentMemberInfo {
    name: string;
    shortDescription: string;
    longDescription: string;
    alias: NamedObjectId[];
    type: ArgumentTypeInfo;
}

interface AlarmInfo {
    minViolations: number;
    staticAlarmRange: AlarmRange[];
    enumerationAlarm: EnumerationAlarm[];
}

interface AlarmRange {
    level: AlarmLevelType;
    minInclusive: number;
    maxInclusive: number;
    minExclusive: number;
    maxExclusive: number;
}

interface EnumerationAlarm {
    level: AlarmLevelType;
    label: string;
}

interface AbsoluteTimeInfo {
    initialValue: string;
    scale: number;
    offset: number;
    offsetFrom: ParameterInfo;
    epoch: string;
}

interface ContextAlarmInfo {
    comparison: ComparisonInfo[];
    alarm: AlarmInfo;

    // This can be used in UpdateParameterRequest to pass a context
    // that is parsed on the server, according to the rules in the
    // excel spreadsheet. Either this or a comparison has to be

```

```

// used (not both at the same time)
context: string;
}

interface MemberInfo {
    name: string;
    shortDescription: string;
    longDescription: string;
    alias: NamedObjectId[];
    type: ParameterTypeInfo;
}

interface ArrayInfo {
    type: ParameterTypeInfo;
    dimensions: ParameterDimensionInfo[];
}

interface ParameterDimensionInfo {
    fixedValue: string; // String decimal
    parameter: ParameterInfo;
    slope: string; // String decimal
    intercept: string; // String decimal
}

interface NumberFormatTypeInfo {
    numberBase: string;
    minimumFractionDigits: number;
    maximumFractionDigits: number;
    minimumIntegerDigits: number;
    maximumIntegerDigits: number;
    negativeSuffix: string;
    positiveSuffix: string;
    negativePrefix: string;
    positivePrefix: string;
    showThousandsGrouping: boolean;
    notation: string;
}

interface UsedByInfo {
    algorithm: AlgorithmInfo[];
    container: ContainerInfo[];
}

interface ContainerInfo {
    name: string;
    qualifiedName: string;
    shortDescription: string;
    longDescription: string;
    alias: NamedObjectId[];
    maxInterval: string; // String decimal
    sizeInBits: number;
    baseContainer: ContainerInfo;
    restrictionCriteria: ComparisonInfo[];
    entry: SequenceEntryInfo[];
    usedBy: UsedByInfo;
    ancillaryData: {[key: string]: AncillaryDataInfo};
}

interface SequenceEntryInfo {
    locationInBits: number;
    referenceLocation: ReferenceLocationType;

    // For use in sequence containers
    container: ContainerInfo;
    parameter: ParameterInfo;

    // For use in command containers
    argument: ArgumentInfo;
    fixedValue: FixedValueInfo;
    repeat: RepeatInfo;
}

```



```

interface FixedValueInfo {
    name: string;
    hexValue: string;
    sizeInBits: number;
}

interface RepeatInfo {
    fixedCount: string; // String decimal
    dynamicCount: ParameterInfo;
    bitsBetween: number;
}

interface OutputParameterInfo {
    parameter: ParameterInfo;
    outputName: string;
}

enum Scope {
    GLOBAL = "GLOBAL",
    COMMAND_VERIFICATION = "COMMAND_VERIFICATION",
    CONTAINER_PROCESSING = "CONTAINER_PROCESSING",
}

enum Type {
    BINARY = "BINARY",
    BOOLEAN = "BOOLEAN",
    FLOAT = "FLOAT",
    INTEGER = "INTEGER",
    STRING = "STRING",
}

enum Type {
    POLYNOMIAL = "POLYNOMIAL",
    SPLINE = "SPLINE",
    MATH_OPERATION = "MATH_OPERATION",
    JAVA_EXPRESSION = "JAVA_EXPRESSION",
}

enum OperatorType {
    EQUAL_TO = "EQUAL_TO",
    NOT_EQUAL_TO = "NOT_EQUAL_TO",
    GREATER_THAN = "GREATER_THAN",
    GREATER_THAN_OR_EQUAL_TO = "GREATER_THAN_OR_EQUAL_TO",
    SMALLER_THAN = "SMALLER_THAN",
    SMALLER_THAN_OR_EQUAL_TO = "SMALLER_THAN_OR_EQUAL_TO",
}

enum AlarmLevelType {
    NORMAL = "NORMAL",
    WATCH = "WATCH",
    WARNING = "WARNING",
    DISTRESS = "DISTRESS",
    CRITICAL = "CRITICAL",
    SEVERE = "SEVERE",
}

enum AlarmLevelType {
    NORMAL = "NORMAL",
    WATCH = "WATCH",
    WARNING = "WARNING",
    DISTRESS = "DISTRESS",
    CRITICAL = "CRITICAL",
    SEVERE = "SEVERE",
}

enum DataSourceType {
    TELEMETERED = "TELEMETERED",
    DERIVED = "DERIVED",
    CONSTANT = "CONSTANT",
    LOCAL = "LOCAL",
}

```

(continued from previous page)

```
SYSTEM = "SYSTEM",  
COMMAND = "COMMAND",  
COMMAND_HISTORY = "COMMAND_HISTORY",  
EXTERNAL1 = "EXTERNAL1",  
EXTERNAL2 = "EXTERNAL2",  
EXTERNAL3 = "EXTERNAL3",  
}  
  
enum ReferenceLocationType {  
    CONTAINER_START = "CONTAINER_START",  
    PREVIOUS_ENTRY = "PREVIOUS_ENTRY",  
}
```

18. Packets

18.1 List Packet Names

List packet names

URI Template

```
GET /api/archive/{instance}/packet-names
```

{instance}

Yamcs instance name.

Response Type

```
interface ListPacketNamesResponse {  
    // Packet name.  
    name: string[];  
}
```

18.2 List Packets

List packets

URI Template

```
GET /api/archive/{instance}/packets
```

{instance}

Yamcs instance name.

Query Parameters

pos

The zero-based row number at which to start outputting results. Default: 0

limit

The maximum number of returned records per page. Choose this value too high and you risk hitting the maximum response size limit enforced by the server. Default: 100

order

The order of the returned results. Can be either asc or desc. Default: desc

name

The archived name of the packets. Names must match exactly.

next

Continuation token returned by a previous page response.

start

Filter the lower bound of the packet's generation time. Specify a date string in ISO 8601 format. This bound is inclusive.

stop

Filter the upper bound of the packet's generation time. Specify a date string in ISO 8601 format. This bound is exclusive.

Response Type

```
interface ListPacketsResponse {
    packet: TmPacketData[];

    // Token indicating the response is only partial. More results can then
    // be obtained by performing the same request (including all original
    // query parameters) and setting the ``next`` parameter to this token.
    continuationToken: string;
}
```

Related Types

```
interface TmPacketData {
    packet: string; // Base64
    sequenceNumber: number;
    id: NamedObjectId;
    receptionTime: string; // RFC 3339 timestamp
    generationTime: string; // RFC 3339 timestamp
}

// Used by external clients to identify an item in the Mission Database
// If namespace is set, then the name is that of an alias, rather than
// the qualified name.
interface NamedObjectId {
    name: string;
    namespace: string;
}
```

18.3 Get Packet

Get a packet

URI Template

```
GET /api/archive/{instance}/packets/{gentime}/{seqnum}
```

{instance}

Yamcs instance name.

{gentime}

An exact match of the packet's generation time in ISO 8601 format.

{seqnum}

Yamcs-specific archive distinguisher

Response Type

```
interface TmPacketData {
    packet: string; // Base64
    sequenceNumber: number;
    id: NamedObjectId;
    receptionTime: string; // RFC 3339 timestamp
    generationTime: string; // RFC 3339 timestamp
}
```

Related Types

```
// Used by external clients to identify an item in the Mission Database
// If namespace is set, then the name is that of an alias, rather than
// the qualified name.
interface NamedObjectId {
    name: string;
    namespace: string;
}
```

18.4 Stream Packets

Streams back packets

Warning: This method uses server-streaming. Yamcs sends an unspecified amount of data using chunked transfer encoding.

URI Template

```
POST /api/stream-archive/{instance}:streamPackets
```

{instance}

Yamcs instance name.

Request Body

```
interface StreamPacketsRequest {
    start: string; // RFC 3339 timestamp
    stop: string; // RFC 3339 timestamp
    name: string[];
}
```

Response Type

```
interface TmPacketData {
  packet: string; // Base64
  sequenceNumber: number;
  id: NamedObjectId;
  receptionTime: string; // RFC 3339 timestamp
  generationTime: string; // RFC 3339 timestamp
}
```

Related Types

```
// Used by external clients to identify an item in the Mission Database
// If namespace is set, then the name is that of an alias, rather than
// the qualified name.
interface NamedObjectId {
  name: string;
  namespace: string;
}
```

18.5 Export Packet

Export a raw packet

URI Template

```
GET /api/archive/{instance}/packets/{gentime}/{seqnum}:export
```

{instance}

Yamcs instance name.

{gentime}

An exact match of the packet's generation time in ISO 8601 format.

{seqnum}

Yamcs-specific archive distinguisher

18.6 Export Packets

Export raw packets

Warning: This method uses server-streaming. Yamcs sends an unspecified amount of data using chunked transfer encoding.

URI Template

```
GET /api/archive/{instance}:exportPackets
```

{instance}

Yamcs instance name.

Query Parameters

start

Filter the lower bound of the packet's generation time. Specify a date string in ISO 8601 format. This bound is inclusive.

stop

Filter the upper bound of the packet's generation time. Specify a date string in ISO 8601 format. This bound is exclusive.

name

The archived name of the packets. Names must match exactly.

18.7 Subscribe Packets

Subscribe to packets

This subscription is performed at stream or processor level.

The identifier of the packets is not filled in.

WebSocket

This method requires to upgrade an HTTP connection to WebSocket. See details on [how Yamcs uses WebSocket¹³](https://docs.yamcs.org/yamcs-http-api/websocket).

Use the message type packets.

Input Type

```
interface SubscribePacketsRequest {  
  
    // Yamcs instance name.  
    instance: string;  
  
    // Stream name. This is mutually exclusive with the field ``processor``.  
    stream: string;  
  
    // Processor name. This is mutually exclusive with the field ``stream``.  
    processor: string;  
}
```

Output Type

```
interface TmPacketData {  
    packet: string; // Base64  
    sequenceNumber: number;  
    id: NamedObjectId;  
    receptionTime: string; // RFC 3339 timestamp  
    generationTime: string; // RFC 3339 timestamp  
}
```

¹³ <https://docs.yamcs.org/yamcs-http-api/websocket>

Related Types

```
// Used by external clients to identify an item in the Mission Database
// If namespace is set, then the name is that of an alias, rather than
// the qualified name.
interface NamedObjectId {
    name: string;
    namespace: string;
}
```

18.8 Subscribe Containers

Subscribe to containers

WebSocket

This method requires to upgrade an HTTP connection to WebSocket. See details on [how Yamcs uses WebSocket¹⁴](#).

Use the message type containers.

Input Type

```
interface SubscribeContainersRequest {

    // Yamcs instance name.
    instance: string;

    // Processor name.
    processor: string;

    // Container names to subscribe to.
    names: string[];
}
```

Output Type

```
interface ContainerData {

    // Container name.
    name: string;

    // When the container's packet was generated (packet time)
    generationTime: string; // RFC 3339 timestamp

    // When the container's packet was received by Yamcs
    receptionTime: string; // RFC 3339 timestamp

    // Container bytes
    binary: string; // Base64
}
```

¹⁴ <https://docs.yamcs.org/yamcs-http-api/websocket>

19. Parameter Archive

19.1 Rebuild Range

Rebuild range

The back filler has to be enabled for this purpose. The back filling process does not remove data but just overwrites it. That means that if the parameter replay returns less parameters than originally stored in the archive, the old parameters will still be found in the archive.

It also means that if a replay returns the parameter of a different type than originally stored, the old ones will still be stored. This is because the parameter archive treats parameter with the same name but different type as different parameters. Each of them is given an id and the id is stored in the archive.

URI Template

```
POST /api/archive/{instance}/parameterArchive:rebuild
```

{instance}

Yamcs instance name.

Request Body

```
// Note that the archive is built in segments of approximately 70 minutes, therefore the
// real start will be before the specified start and the real stop will be after the
// specified stop.
interface RebuildRangeRequest {

    // Start rebuilding from here. Specify a date string in ISO 8601 format.
    start: string; // RFC 3339 timestamp

    // Rebuild until here. Specify a date string in ISO 8601 format.
    stop: string; // RFC 3339 timestamp
}
```

19.2 Delete Partitions

Delete partitions

Response is of type string and list the partitions that have been removed.

URI Template

```
POST /api/archive/{instance}/parameterArchive:deletePartitions
```

{instance}
Yamcs instance name.

Request Body

```
interface DeletePartitionsRequest {  
  
    // Start with the partition that contains this timestamp. Specify a date string in ISO 8601 format.  
    start: string; // RFC 3339 timestamp  
  
    // Stop with the partition that contains this timestamp. The stop partition will be removed as  
    // well. Specify a date string in ISO 8601 format.  
    stop: string; // RFC 3339 timestamp  
}
```

Response Type

```
interface StringMessage {  
    message: string;  
}
```

19.3 Get Parameter Samples

Get parameter samples

This divides the query interval in a number of intervals and returns aggregated statistics (max,min,avg) about each interval.

This operation is useful when making high-level overviews (such as plots) of a parameter's value over large time intervals without having to retrieve each and every individual parameter value.

By default this operation fetches data from the parameter archive and/or parameter cache. If these services are not configured, you can still get correct results by specifying the option `source=replay` as detailed below.

URI Template

```
GET /api/archive/{instance}/parameters/{name*}/samples
```

{instance}
Yamcs instance name.

{name*}
Parameter name.

Query Parameters

start

Filter the lower bound of the parameter's generation time. Specify a date string in ISO 8601 format.

stop

Filter the upper bound of the parameter's generation time. Specify a date string in ISO 8601 format.

count

Number of intervals to use. Default: 500.

norealtime

Disable loading of parameters from the parameter cache. Default: false.

useRawValue

Consider the raw value instead of the engineering value. Default is to use the engineering value

processor

The name of the processor from which to use the parameter cache. Default: realtime.

source

Specifies how to retrieve the parameters. Either ParameterArchive or replay. If replay is specified, a replay processor will be created and data will be processed with the active Mission Database. Note that this is much slower than receiving data from the ParameterArchive.

Default: ParameterArchive.

Response Type

```
interface TimeSeries {  
  sample: Sample[];  
}
```

Related Types

```
interface Sample {  
  time: string;  
  avg: number;  
  min: number;  
  max: number;  
  n: number;  
}
```

19.4 Get Parameter Ranges

Get parameter ranges

A range is a tuple (start, stop, value, count) that represents the time interval for which the parameter has been steadily coming in with the same value. This request is useful for retrieving an overview for parameters that change unfrequently in a large time interval. For example an on/off status of a device, or some operational status. Two consecutive ranges containing the same value will be returned if there was a gap in the data. The gap is determined according to the parameter expiration time configured in the Mission Database.

URI Template

```
GET /api/archive/{instance}/parameters/{name*}/ranges
```

{instance}

Yamcs instance name.

{name*}

Parameter name.

Query Parameters

start

Filter the lower bound of the parameter's generation time. Specify a date string in ISO 8601 format.

stop

Filter the upper bound of the parameter's generation time. Specify a date string in ISO 8601 format.

minGap

Time in milliseconds. Any gap (detected based on parameter expiration) smaller than this will be ignored. However if the parameter changes value, the ranges will still be split.

maxGap

Time in milliseconds. If the distance between two subsequent values of the parameter is bigger than this value (but smaller than the parameter expiration), then an artificial gap will be constructed. This also applies if there is no parameter expiration defined for the parameter.

norealtime

Disable loading of parameters from the parameter cache. Default: `false`.

processor

The name of the processor from which to use the parameter cache. Default: `realtime`.

source

minRange

Time in milliseconds of the minimum range to be returned. If the data changes more often, a new range will not be created but the data will be added to the old range.

maxValues

Maximum number of distinct values to be returned. The maximum number applies across all ranges and is meant to limit the amount of data that is being retrieved. The retrieved data has a count for each value as well as a total count. The difference between the total count and the sum of the individual counts can be used to compute the number of unsent values.

Response Type

```
interface Ranges {  
    range: Range[];  
}
```

Related Types

```
interface Range {  
  
    // Generation time of a parameter value.  
    start: string; // RFC 3339 timestamp  
  
    // If the value changes, ``stop`` is the generation time of the new value.  
    // If the parameter expires or the ``maxGap`` has been set, ``stop`` is  
    // the generation time of the last value plus the expiration time or the  
    // ``maxGap``.  
    stop: string; // RFC 3339 timestamp  
  
    // Deprecated. Use ``start`` instead.  
    timeStart: string;  
  
    // Deprecated. Use ``stop`` instead.  
    timeStop: string;  
  
    // Number of parameter values received in the interval.  
    // This is the total count of parameters in the interval.  
    // If the count does not match the sum(counts), it means that not all the values have been sent  
    count: number;  
  
    // Since Yamcs 5.4.1 there is a new parameter minRange in the GetParameterRangesRequest which allows  
    // specifying the minimum length of the range returned.  
    // Practically we guarantee that stop-start >= minRange (mind the leap seconds!).  
    //  
    // If the minRange parameter is set, the returning ranges may include multiple values.  
    // These are given by the engValues and counts below.  
    //  
    // Since Yamcs 5.4.2 there is a new parameter maxValues which allows to limit the number  
    // of distinct values returned across all the ranges.  
    // In order to not return ranges containing no value, each range will have at least one value even if  
    // that will cause the total number of range values returned to exceed the maxValues parameter  
    //  
    // The counts correspond one to one to the engValues, the two arrays will always have the same length.  
    engValues: Value[];  
  
    // The counts correspond one to one to the engValues  
    counts: number[];  
}  
  
// Union type for storing a value  
interface Value {  
    type: Type;  
    floatValue: number;  
    doubleValue: number;  
    sint32Value: number;  
    uint32Value: number;  
    binaryValue: string; // Base64  
    stringValue: string;  
    timestampValue: string; // String decimal  
    uint64Value: string; // String decimal  
    sint64Value: string; // String decimal  
    booleanValue: boolean;  
    aggregateValue: AggregateValue;  
    arrayValue: Value[];  
}  
  
// An aggregate value is an ordered list of (member name, member value).  
// Two arrays are used in order to be able to send just the values (since  
// the names will not change)
```

(continues on next page)

(continued from previous page)

```
interface AggregateValue {
    name: string[];
    value: Value[];
}

enum Type {
    FLOAT = "FLOAT",
    DOUBLE = "DOUBLE",
    UINT32 = "UINT32",
    SINT32 = "SINT32",
    BINARY = "BINARY",
    STRING = "STRING",
    TIMESTAMP = "TIMESTAMP",
    UINT64 = "UINT64",
    SINT64 = "SINT64",
    BOOLEAN = "BOOLEAN",
    AGGREGATE = "AGGREGATE",
    ARRAY = "ARRAY",

    // Enumerated values have both an integer (sint64Value) and a string representation
    ENUMERATED = "ENUMERATED",
    NONE = "NONE",
}
```

19.5 List Parameter History

List parameter history

URI Template

```
GET /api/archive/{instance}/parameters/{name*}
```

{instance}

Yamcs instance name.

{name*}

Parameter name.

Query Parameters

pos

The zero-based row number at which to start outputting results. Default: 0.

limit

The maximum number of returned records per page. Choose this value too high and you risk hitting the maximum response size limit enforced by the server. Default: 100.

norepeat

Whether to filter out consecutive identical values. Default no.

start

Filter the lower bound of the parameter's generation time. Specify a date string in ISO 8601 format.

stop

Filter the upper bound of the parameter's generation time. Specify a date string in ISO 8601 format.

order

The order of the returned results. Can be either asc or desc. Default: desc.

norealtime

Disable loading of parameters from the parameter cache. Default: false.

processor

The name of the processor from which to use the parameter cache. Default: realtime.

source

Specifies how to retrieve the parameters. Either ParameterArchive or replay. If replay is specified, a replay processor will be created and data will be processed with the active Mission Database. Note that this is much slower than receiving data from the ParameterArchive.

Default: ParameterArchive.

next

Continuation token returned by a previous page response.

Response Type

```
interface ListParameterHistoryResponse {
    parameter: ParameterValue[];

    // Token indicating the response is only partial. More results can then
    // be obtained by performing the same request (including all original
    // query parameters) and setting the ``next`` parameter to this token.
    continuationToken: string;
}
```

Related Types

```
interface ParameterValue {

    // Parameter identifier
    id: NamedObjectId;

    // Raw value (uncalibrated)
    rawValue: Value;

    // Engineering value (calibrated)
    engValue: Value;

    // Time of Yamcs reception
    acquisitionTime: string; // RFC 3339 timestamp

    // Time of generation (~ packet time)
    generationTime: string; // RFC 3339 timestamp
    acquisitionStatus: AcquisitionStatus;

    // Deprecated: this field was originally introduced for compatibility
```

(continues on next page)

```

// with Airbus CGS/CD-MCS system. It was redundant, because when false,
// the acquisitionStatus is also set to INVALID.
processingStatus: boolean;
monitoringResult: MonitoringResult;
rangeCondition: RangeCondition;

// Context-dependent ranges
alarmRange: AlarmRange[];

// How long (in milliseconds) this parameter value is valid
// Note that there is an option when subscribing to parameters to get
// updated when the parameter values expire.
expireMillis: string; // String decimal

// When transferring parameters over WebSocket, this value might be used
// instead of the id above in order to reduce the bandwidth.
// Note that the id <-> numericId assignment is only valid in the context
// of a single WebSocket call.
numericId: number;
}

// Used by external clients to identify an item in the Mission Database
// If namespace is set, then the name is that of an alias, rather than
// the qualified name.
interface NamedObjectId {
    name: string;
    namespace: string;
}

// Union type for storing a value
interface Value {
    type: Type;
    floatValue: number;
    doubleValue: number;
    sint32Value: number;
    uint32Value: number;
    binaryValue: string; // Base64
    stringValue: string;
    timestampValue: string; // String decimal
    uint64Value: string; // String decimal
    sint64Value: string; // String decimal
    booleanValue: boolean;
    aggregateValue: AggregateValue;
    arrayValue: Value[];
}

// An aggregate value is an ordered list of (member name, member value).
// Two arrays are used in order to be able to send just the values (since
// the names will not change)
interface AggregateValue {
    name: string[];
    value: Value[];
}

interface AlarmRange {
    level: AlarmLevelType;
    minInclusive: number;
    maxInclusive: number;
    minExclusive: number;
    maxExclusive: number;
}

enum Type {
    FLOAT = "FLOAT",
    DOUBLE = "DOUBLE",
    UINT32 = "UINT32",
    SINT32 = "SINT32",
    BINARY = "BINARY",
    STRING = "STRING",
    TIMESTAMP = "TIMESTAMP",
    UINT64 = "UINT64",

```



```

SINT64 = "SINT64",
BOOLEAN = "BOOLEAN",
AGGREGATE = "AGGREGATE",
ARRAY = "ARRAY",

// Enumerated values have both an integer (sint64Value) and a string representation
ENUMERATED = "ENUMERATED",
NONE = "NONE",
}

enum AcquisitionStatus {

    // OK!
    ACQUIRED = "ACQUIRED",

    // No value received so far
    NOT_RECEIVED = "NOT_RECEIVED",

    // Some value has been received but is invalid
    INVALID = "INVALID",

    // The parameter is coming from a packet which has not since updated although it should have been
    EXPIRED = "EXPIRED",
}

enum MonitoringResult {
    DISABLED = "DISABLED",
    IN_LIMITS = "IN_LIMITS",
    WATCH = "WATCH",
    WARNING = "WARNING",
    DISTRESS = "DISTRESS",
    CRITICAL = "CRITICAL",
    SEVERE = "SEVERE",
}

enum RangeCondition {
    LOW = "LOW",
    HIGH = "HIGH",
}

enum AlarmLevelType {
    NORMAL = "NORMAL",
    WATCH = "WATCH",
    WARNING = "WARNING",
    DISTRESS = "DISTRESS",
    CRITICAL = "CRITICAL",
    SEVERE = "SEVERE",
}

```

19.6 Get Archived Parameters Info

Get information about the archived parameters.

Each combination of (parameter name, raw type, engineering type) is assigned a unique parameter id.

The parameters are grouped such that the samples of all parameters from one group have the same timestamp. For example all parameters extracted from one TM packet have usually the same timestamp and are part of the same group.

Each group is assigned a unique group id.

A parameter can be part of multiple groups - for instance a parameter appearing in the header of a packet is part of all groups made by inherited containers (i.e. each packet with that header will compose another group).

For each group, the parameter archive stores one common record for the timestamps and individual records for the raw and engineering values of each parameter. If a parameter appears in multiple groups, retrieving

its value means combining (time based merge operation) the records belonging to the groups in which the parameter appears.

The response to this method contains the parameter id, name, engineering type, raw type and the groups of which this parameter is part of.

URI Template

```
GET /api/archive/{instance}/parameterArchive/info/parameters
```

{instance}

Query Parameters

q

The search keywords.

system

List only direct child parameters of the specified system. Only the parameters whose fully qualified name start with system will be returned.

When system and q are used together, the q search will be matched on the parameters filtered by system.

limit

The maximum number of returned parameters. Choose this value too high and you risk hitting the maximum response size limit enforced by the server. Default: 100

Response Type

```
interface ArchivedParametersInfoResponse {  
    parameters: ArchivedParameterInfo[];  
}
```

Related Types

```
// This message contains information about one parameter in the parameter archive  
// Each (parameter name, raw type, engineering type) is assigned a unique id and all  
// the samples are stored with that id.  
// If a MDB change results in the parameter having a different engineering or raw type,  
// a new pid will be allocated.  
// This is why for the same parameter name, we can have multiple parameter ids.  
// The parameter archive will contain data even for parameters removed from the MDB  
interface ArchivedParameterInfo {
```

```
    //parameter id  
    pid: number;
```

```
    //parameter fully qualified name  
    fqcn: string;
```

```
    // parameter raw type  
    rawType: Type;
```

(continues on next page)

```

//parameter engineering type
engType: Type;

//the groups from which this parameter is part of
gids: number[];
}

enum Type {
    FLOAT = "FLOAT",
    DOUBLE = "DOUBLE",
    UINT32 = "UINT32",
    SINT32 = "SINT32",
    BINARY = "BINARY",
    STRING = "STRING",
    TIMESTAMP = "TIMESTAMP",
    UINT64 = "UINT64",
    SINT64 = "SINT64",
    BOOLEAN = "BOOLEAN",
    AGGREGATE = "AGGREGATE",
    ARRAY = "ARRAY",

    // Enumerated values have both an integer (sint64Value) and a string representation
    ENUMERATED = "ENUMERATED",
    NONE = "NONE",
}

```

19.7 Get Archived Parameter Segments

For a given parameter id, get the list of segments available for that parameter. A segment contains multiple samples (maximum ~70 minutes) of the same parameter

URI Template

```
GET /api/archive/{instance}/parameterArchive/info/segments/{pid*}
```

{instance}
Yamcs instance

{pid*}
parameter id

Query Parameters

start

get the segments overlapping with [start, stop) interval

stop

Response Type

```

// Recorded segments for the requested parameter
interface ArchivedParameterSegmentsResponse {
    parameterInfo: ArchivedParameterInfo;
    segments: ArchiveParameterSegmentInfo[];
}

```

Related Types

```
// This message contains information about one parameter in the parameter archive
// Each (parameter name, raw type, engineering type) is assigned a unique id and all
// the samples are stored with that id.
// If a MDB change results in the parameter having a different engineering or raw type,
// a new pid will be allocated.
// This is why for the same parameter name, we can have multiple parameter ids.
// The parameter archive will contain data even for parameters removed from the MDB
interface ArchivedParameterInfo {

    //parameter id
    pid: number;

    //parameter fully qualified name
    fqcn: string;

    // parameter raw type
    rawType: Type;

    //parameter engineering type
    engType: Type;

    //the groups from which this parameter is part of
    gids: number[];
}

interface ArchiveParameterSegmentInfo {

    // Multiple parameters are grouped such that all in one group have
    // the same timestamps. For example: all parameters extracted from
    // one TM packet usually have the same timestamp.
    // This way we have a unique segment storing the timestamps for a
    // group of parameters. The groupId can be used to retrieve all parameters
    // from the same group.
    groupId: number;

    //the segment start
    start: string; // RFC 3339 timestamp

    //the segment end
    end: string; // RFC 3339 timestamp

    //the number of samples in the segment
    count: number;
}

enum Type {
    FLOAT = "FLOAT",
    DOUBLE = "DOUBLE",
    UINT32 = "UINT32",
    SINT32 = "SINT32",
    BINARY = "BINARY",
    STRING = "STRING",
    TIMESTAMP = "TIMESTAMP",
    UINT64 = "UINT64",
    SINT64 = "SINT64",
    BOOLEAN = "BOOLEAN",
    AGGREGATE = "AGGREGATE",
    ARRAY = "ARRAY",

    // Enumerated values have both an integer (sint64Value) and a string representation
    ENUMERATED = "ENUMERATED",
    NONE = "NONE",
}
```

19.8 Get Archived Parameter Group

For a given group id, get the list of parameters which are part of the group

URI Template

```
GET /api/archive/{instance}/parameterArchive/info/groups/{gid*}
```

{instance}

Yamcs instance name

{gid*}

Group identifier

Response Type

```
interface ArchivedParameterGroupResponse {  
  
    // Group identifier  
    gid: number;  
  
    // Parameters belonging to the group  
    parameters: ArchivedParameterInfo[];  
}
```

Related Types

```
// This message contains information about one parameter in the parameter archive  
// Each (parameter name, raw type, engineering type) is assigned a unique id and all  
// the samples are stored with that id.  
// If a MDB change results in the parameter having a different engineering or raw type,  
// a new pid will be allocated.  
// This is why for the same parameter name, we can have multiple parameter ids.  
// The parameter archive will contain data even for parameters removed from the MDB  
interface ArchivedParameterInfo {  
  
    //parameter id  
    pid: number;  
  
    //parameter fully qualified name  
    fqcn: string;  
  
    // parameter raw type  
    rawType: Type;  
  
    //parameter engineering type  
    engType: Type;  
  
    //the groups from which this parameter is part of  
    gids: number[];  
}  
  
enum Type {  
    FLOAT = "FLOAT",  
    DOUBLE = "DOUBLE",  
    UINT32 = "UINT32",  
    SINT32 = "SINT32",  
    BINARY = "BINARY",  
    STRING = "STRING",  
    TIMESTAMP = "TIMESTAMP",  
    UINT64 = "UINT64",  
    SINT64 = "SINT64",  
    BOOLEAN = "BOOLEAN",  
    AGGREGATE = "AGGREGATE",  
    ARRAY = "ARRAY",  
  
    // Enumerated values have both an integer (sint64Value) and a string representation  
    ENUMERATED = "ENUMERATED",  
    NONE = "NONE",  
}
```

20. Processing

20.1 List Processor Types

List processor types

URI Template

```
GET /api/processor-types
```

Response Type

```
interface ListProcessorTypesResponse {  
    types: string[];  
}
```

20.2 List Processors

List processors

URI Template

```
GET /api/processors
```

Query Parameters

instance

Return only processors of this instance

Response Type

```
interface ListProcessorsResponse {  
    processors: ProcessorInfo[];  
}
```

Related Types

```
interface ProcessorInfo {  
  
    // Yamcs instance name.  
    instance: string;  
  
    // Processor name.  
    name: string;  
    type: string;  
    spec: string;  
    creator: string;  
    hasAlarms: boolean;  
    hasCommanding: boolean;  
    state: ServiceState;  
    replayRequest: ReplayRequest;  
    replayState: ReplayState;  
    services: ServiceInfo[];  
    persistent: boolean;  
    time: string; // RFC 3339 timestamp  
    replay: boolean;  
    checkCommandClearance: boolean;  
}  
  
//used to replay (concurrently) TM packets, parameters and events  
interface ReplayRequest {  
  
    // **Required.** The time at which the replay should start.  
    start: string; // RFC 3339 timestamp  
  
    // The time at which the replay should stop.  
    // If unspecified, the replay will keep going as long as there is remaining data.  
    stop: string; // RFC 3339 timestamp  
  
    //what should happen at the end of the replay  
    endAction: EndAction;  
  
    //how fast the replay should go  
    speed: ReplaySpeed;  
  
    // Reverse the direction of the replay  
    reverse: boolean;  
    parameterRequest: ParameterReplayRequest;  
  
    // By default all Packets, Events, CommandHistory are part of the replay  
    // Unless one or more of the below requests are specified.  
    packetRequest: PacketReplayRequest;  
    eventRequest: EventReplayRequest;  
    commandHistoryRequest: CommandHistoryReplayRequest;  
    ppRequest: PpReplayRequest;  
}  
  
interface ReplaySpeed {  
    type: ReplaySpeedType;  
    param: number;  
}  
  
interface ParameterReplayRequest {  
    nameFilter: NamedObjectId[];  
    sendRaw: boolean;  
    performMonitoring: boolean;  
}  
  
// Used by external clients to identify an item in the Mission Database  
// If namespace is set, then the name is that of an alias, rather than  
// the qualified name.  
interface NamedObjectId {  
    name: string;  
    namespace: string;  
}
```

(continues on next page)

```

interface PacketReplayRequest {

    // No filter, means all packets for which privileges exist, are sent
    nameFilter: NamedObjectId[];
}

interface EventReplayRequest {
}

interface CommandHistoryReplayRequest {

    // No filter, means all command history entries are sent
    nameFilter: NamedObjectId[];
}

//Request to replay parameters - they can be filtered by the parameter group
interface PpReplayRequest {

    // No filter, means all pp groups are sent
    groupNameFilter: string[];

    // exclude the parameters from these groups
    // this takes precedence over the filter above (i.e. if a group is part of both, it will be excluded)
    groupNameExclude: string[];
}

interface ServiceInfo {
    instance: string;
    name: string;
    state: ServiceState;
    className: string;
    processor: string;
}

enum ServiceState {
    NEW = "NEW",
    STARTING = "STARTING",
    RUNNING = "RUNNING",
    STOPPING = "STOPPING",
    TERMINATED = "TERMINATED",
    FAILED = "FAILED",
}

enum EndAction {
    LOOP = "LOOP",
    QUIT = "QUIT",
    STOP = "STOP",
}

enum ReplaySpeedType {
    AFAP = "AFAP",
    FIXED_DELAY = "FIXED_DELAY",
    REALTIME = "REALTIME",
    STEP_BY_STEP = "STEP_BY_STEP",
}

enum ReplayState {

    // just at the beginning or when the replay request (start, stop or packet selection) changes
    INITIALIZATION = "INITIALIZATION",
    RUNNING = "RUNNING",

    // The replay has reached the end with the endaction stop
    STOPPED = "STOPPED",

    // The replay stopped due to an error.
    ERROR = "ERROR",
    PAUSED = "PAUSED",

    // The replay is finished and closed
    CLOSED = "CLOSED",
}

```


(continued from previous page)

```
}

enum ServiceState {
    NEW = "NEW",
    STARTING = "STARTING",
    RUNNING = "RUNNING",
    STOPPING = "STOPPING",
    TERMINATED = "TERMINATED",
    FAILED = "FAILED",
}
```

20.3 Get Processor

Get a processor

URI Template

```
GET /api/processors/{instance}/{processor}
```

{instance}
Yamcs instance name.

{processor}
Processor name.

Response Type

```
interface ProcessorInfo {

    // Yamcs instance name.
    instance: string;

    // Processor name.
    name: string;
    type: string;
    spec: string;
    creator: string;
    hasAlarms: boolean;
    hasCommanding: boolean;
    state: ServiceState;
    replayRequest: ReplayRequest;
    replayState: ReplayState;
    services: ServiceInfo[];
    persistent: boolean;
    time: string; // RFC 3339 timestamp
    replay: boolean;
    checkCommandClearance: boolean;
}
```

Related Types

```
//used to replay (concurrently) TM packets, parameters and events
interface ReplayRequest {

    // **Required.** The time at which the replay should start.
    start: string; // RFC 3339 timestamp

    // The time at which the replay should stop.
}
```

(continues on next page)

(continued from previous page)

```
// If unspecified, the replay will keep going as long as there is remaining data.
stop: string; // RFC 3339 timestamp

//what should happen at the end of the replay
endAction: EndAction;

//how fast the replay should go
speed: ReplaySpeed;

// Reverse the direction of the replay
reverse: boolean;
parameterRequest: ParameterReplayRequest;

// By default all Packets, Events, CommandHistory are part of the replay
// Unless one or more of the below requests are specified.
packetRequest: PacketReplayRequest;
eventRequest: EventReplayRequest;
commandHistoryRequest: CommandHistoryReplayRequest;
ppRequest: PpReplayRequest;
}

interface ReplaySpeed {
    type: ReplaySpeedType;
    param: number;
}

interface ParameterReplayRequest {
    nameFilter: NamedObjectId[];
    sendRaw: boolean;
    performMonitoring: boolean;
}

// Used by external clients to identify an item in the Mission Database
// If namespace is set, then the name is that of an alias, rather than
// the qualified name.
interface NamedObjectId {
    name: string;
    namespace: string;
}

interface PacketReplayRequest {

    // No filter, means all packets for which privileges exist, are sent
    nameFilter: NamedObjectId[];
}

interface EventReplayRequest {
}

interface CommandHistoryReplayRequest {

    // No filter, means all command history entries are sent
    nameFilter: NamedObjectId[];
}

//Request to replay parameters - they can be filtered by the parameter group
interface PpReplayRequest {

    // No filter, means all pp groups are sent
    groupNameFilter: string[];

    // exclude the parameters from these groups
    // this takes precedence over the filter above (i.e. if a group is part of both, it will be excluded)
    groupNameExclude: string[];
}

interface ServiceInfo {
    instance: string;
    name: string;
    state: ServiceState;
    className: string;
}
```

(continues on next page)

```

processor: string;
}

enum ServiceState {
    NEW = "NEW",
    STARTING = "STARTING",
    RUNNING = "RUNNING",
    STOPPING = "STOPPING",
    TERMINATED = "TERMINATED",
    FAILED = "FAILED",
}

enum EndAction {
    LOOP = "LOOP",
    QUIT = "QUIT",
    STOP = "STOP",
}

enum ReplaySpeedType {
    AFAP = "AFAP",
    FIXED_DELAY = "FIXED_DELAY",
    REALTIME = "REALTIME",
    STEP_BY_STEP = "STEP_BY_STEP",
}

enum ReplayState {

    // just at the beginning or when the replay request (start, stop or packet selection) changes
    INITIALIZATION = "INITIALIZATION",
    RUNNING = "RUNNING",

    // The replay has reached the end with the endaction stop
    STOPPED = "STOPPED",

    // The replay stopped due to an error.
    ERROR = "ERROR",
    PAUSED = "PAUSED",

    // The replay is finished and closed
    CLOSED = "CLOSED",
}

enum ServiceState {
    NEW = "NEW",
    STARTING = "STARTING",
    RUNNING = "RUNNING",
    STOPPING = "STOPPING",
    TERMINATED = "TERMINATED",
    FAILED = "FAILED",
}

```

20.4 Delete Processor

Delete a processor

Only replay processors can be removed.

URI Template

```
DELETE /api/processors/{instance}/{processor}
```

{instance}

Yamcs instance name.

{processor}
Processor name.

20.5 Edit Processor

Update a processor

URI Template

```
PATCH /api/processors/{instance}/{processor}
```

{instance}
Yamcs instance name.

{processor}
Processor name.

Request Body

```
interface EditProcessorRequest {  
  
    // The state this replay processor should be updated to. Either `paused` or  
    // `running`.  
    state: string;  
  
    // The time where the processing needs to jump towards. Must be a date string  
    // in ISO 8601 format.  
    seek: string; // RFC 3339 timestamp  
  
    // The speed of the processor. One of:  
    // * `afap`  
    // * a speed factor relative to the original speed. Example: `2x`  
    // * a fixed delay value in milliseconds. Example: `2000`  
    speed: string;  
}
```

20.6 Create Processor

Create a processor

URI Template

```
POST /api/processors
```

Request Body

```
interface CreateProcessorRequest {  
  
    // Required. The name of the Yamcs instance.  
    instance: string;  
  
    // Required. The name of the processor. Must be unique for the Yamcs instance.  
    name: string;  
}
```

(continues on next page)

(continued from previous page)

```
// Keep the processor when terminated. Default: ``no``.
persistent: boolean;

// **Required.** The type of the processor. The available values depend on how
// Yamcs Server is configured. Most Yamcs deployments support at least a type
// ``Archive`` which allows for the creation of processors replaying archived
// data.
type: string;

// Configuration options specific to the processor type. Note that this should
// be a string representation of a valid JSON structure.
config: string;
}
```

20.7 Get Parameter Value

Get a parameter's value

URI Template

```
GET /api/processors/{instance}/{processor}/parameters/{name*}
```

{instance}
Yamcs instance name.

{processor}
Processor name.

{name*}
Parameter name.

Query Parameters

fromCache

Whether the latest cached value may be returned. Default: yes.

timeout

Time in milliseconds to wait on a value (only considered if `fromCache=no`). When the timeout is met, the call will return with no or partial data. Default: 10000.

Response Type

```
interface ParameterValue {

    // Parameter identifier
    id: NamedObjectId;

    // Raw value (uncalibrated)
    rawValue: Value;

    // Engineering value (calibrated)
    engValue: Value;

    // Time of Yamcs reception
}
```

(continues on next page)

```

acquisitionTime: string; // RFC 3339 timestamp

// Time of generation (~ packet time)
generationTime: string; // RFC 3339 timestamp
acquisitionStatus: AcquisitionStatus;

// Deprecated: this field was originally introduced for compatibility
// with Airbus CGS/CD-MCS system. It was redundant, because when false,
// the acquisitionStatus is also set to INVALID.
processingStatus: boolean;
monitoringResult: MonitoringResult;
rangeCondition: RangeCondition;

// Context-dependent ranges
alarmRange: AlarmRange[];

// How long (in milliseconds) this parameter value is valid
// Note that there is an option when subscribing to parameters to get
// updated when the parameter values expire.
expireMillis: string; // String decimal

// When transferring parameters over WebSocket, this value might be used
// instead of the id above in order to reduce the bandwidth.
// Note that the id <-> numericId assignment is only valid in the context
// of a single WebSocket call.
numericId: number;
}

```

Related Types

```

// Used by external clients to identify an item in the Mission Database
// If namespace is set, then the name is that of an alias, rather than
// the qualified name.
interface NamedObjectId {
    name: string;
    namespace: string;
}

// Union type for storing a value
interface Value {
    type: Type;
    floatValue: number;
    doubleValue: number;
    sint32Value: number;
    uint32Value: number;
    binaryValue: string; // Base64
    stringValue: string;
    timestampValue: string; // String decimal
    uint64Value: string; // String decimal
    sint64Value: string; // String decimal
    booleanValue: boolean;
    aggregateValue: AggregateValue;
    arrayValue: Value[];
}

// An aggregate value is an ordered list of (member name, member value).
// Two arrays are used in order to be able to send just the values (since
// the names will not change)
interface AggregateValue {
    name: string[];
    value: Value[];
}

interface AlarmRange {
    level: AlarmLevelType;
    minInclusive: number;
    maxInclusive: number;
    minExclusive: number;
}

```

```

    maxExclusive: number;
}

enum Type {
    FLOAT = "FLOAT",
    DOUBLE = "DOUBLE",
    UINT32 = "UINT32",
    SINT32 = "SINT32",
    BINARY = "BINARY",
    STRING = "STRING",
    TIMESTAMP = "TIMESTAMP",
    UINT64 = "UINT64",
    SINT64 = "SINT64",
    BOOLEAN = "BOOLEAN",
    AGGREGATE = "AGGREGATE",
    ARRAY = "ARRAY",

    // Enumerated values have both an integer (sint64Value) and a string representation
    ENUMERATED = "ENUMERATED",
    NONE = "NONE",
}

enum AcquisitionStatus {

    // OK!
    ACQUIRED = "ACQUIRED",

    // No value received so far
    NOT_RECEIVED = "NOT_RECEIVED",

    // Some value has been received but is invalid
    INVALID = "INVALID",

    // The parameter is coming from a packet which has not since updated although it should have been
    EXPIRED = "EXPIRED",
}

enum MonitoringResult {
    DISABLED = "DISABLED",
    IN_LIMITS = "IN_LIMITS",
    WATCH = "WATCH",
    WARNING = "WARNING",
    DISTRESS = "DISTRESS",
    CRITICAL = "CRITICAL",
    SEVERE = "SEVERE",
}

enum RangeCondition {
    LOW = "LOW",
    HIGH = "HIGH",
}

enum AlarmLevelType {
    NORMAL = "NORMAL",
    WATCH = "WATCH",
    WARNING = "WARNING",
    DISTRESS = "DISTRESS",
    CRITICAL = "CRITICAL",
    SEVERE = "SEVERE",
}

```

20.8 Set Parameter Value

Set a parameter's value

Only some type of parameters can be updated.

URI Template

```
PUT /api/processors/{instance}/{processor}/parameters/{name*}
```

{instance}

Yamcs instance name.

{processor}

Processor name.

{name*}

Parameter name.

Request Body

```
// Union type for storing a value
interface Value {
  type: Type;
  floatValue: number;
  doubleValue: number;
  sint32Value: number;
  uint32Value: number;
  binaryValue: string; // Base64
  stringValue: string;
  timestampValue: string; // String decimal
  uint64Value: string; // String decimal
  sint64Value: string; // String decimal
  booleanValue: boolean;
  aggregateValue: AggregateValue;
  arrayValue: Value[];
}
```

Related Types

```
// An aggregate value is an ordered list of (member name, member value).
// Two arrays are used in order to be able to send just the values (since
// the names will not change)
interface AggregateValue {
  name: string[];
  value: Value[];
}

enum Type {
  FLOAT = "FLOAT",
  DOUBLE = "DOUBLE",
  UINT32 = "UINT32",
  SINT32 = "SINT32",
  BINARY = "BINARY",
  STRING = "STRING",
  TIMESTAMP = "TIMESTAMP",
  UINT64 = "UINT64",
  SINT64 = "SINT64",
  BOOLEAN = "BOOLEAN",
  AGGREGATE = "AGGREGATE",
  ARRAY = "ARRAY",

  // Enumerated values have both an integer (sint64Value) and a string representation
  ENUMERATED = "ENUMERATED",
  NONE = "NONE",
}
```


20.9 Batch Get Parameter Values

Batch get the value of multiple parameters

URI Template

```
POST /api/processors/{instance}/{processor}/parameters:batchGet
```

{instance}

Yamcs instance name.

{processor}

Processor name.

Request Body

```
interface BatchGetParameterValuesRequest {
    id: NamedObjectId[];
    fromCache: boolean;

    // if not fromCache, wait this time (in milliseconds) to receive the parameter
    timeout: string; // String decimal
}
```

Response Type

```
interface BatchGetParameterValuesResponse {
    value: ParameterValue[];
}
```

Related Types

```
// Used by external clients to identify an item in the Mission Database
// If namespace is set, then the name is that of an alias, rather than
// the qualified name.
interface NamedObjectId {
    name: string;
    namespace: string;
}

interface ParameterValue {

    // Parameter identifier
    id: NamedObjectId;

    // Raw value (uncalibrated)
    rawValue: Value;

    // Engineering value (calibrated)
    engValue: Value;

    // Time of Yamcs reception
    acquisitionTime: string; // RFC 3339 timestamp

    // Time of generation (~ packet time)
    generationTime: string; // RFC 3339 timestamp
    acquisitionStatus: AcquisitionStatus;

    // Deprecated: this field was originally introduced for compatibility
    // with Airbus CGS/CD-MCS system. It was redundant, because when false,
```

(continues on next page)

```

// the acquisitionStatus is also set to INVALID.
processingStatus: boolean;
monitoringResult: MonitoringResult;
rangeCondition: RangeCondition;

// Context-dependent ranges
alarmRange: AlarmRange[];

// How long (in milliseconds) this parameter value is valid
// Note that there is an option when subscribing to parameters to get
// updated when the parameter values expire.
expireMillis: string; // String decimal

// When transferring parameters over WebSocket, this value might be used
// instead of the id above in order to reduce the bandwidth.
// Note that the id <-> numericId assignment is only valid in the context
// of a single WebSocket call.
numericId: number;
}

// Union type for storing a value
interface Value {
  type: Type;
  floatValue: number;
  doubleValue: number;
  sint32Value: number;
  uint32Value: number;
  binaryValue: string; // Base64
  stringValue: string;
  timestampValue: string; // String decimal
  uint64Value: string; // String decimal
  sint64Value: string; // String decimal
  booleanValue: boolean;
  aggregateValue: AggregateValue;
  arrayValue: Value[];
}

// An aggregate value is an ordered list of (member name, member value).
// Two arrays are used in order to be able to send just the values (since
// the names will not change)
interface AggregateValue {
  name: string[];
  value: Value[];
}

interface AlarmRange {
  level: AlarmLevelType;
  minInclusive: number;
  maxInclusive: number;
  minExclusive: number;
  maxExclusive: number;
}

enum Type {
  FLOAT = "FLOAT",
  DOUBLE = "DOUBLE",
  UINT32 = "UINT32",
  SINT32 = "SINT32",
  BINARY = "BINARY",
  STRING = "STRING",
  TIMESTAMP = "TIMESTAMP",
  UINT64 = "UINT64",
  SINT64 = "SINT64",
  BOOLEAN = "BOOLEAN",
  AGGREGATE = "AGGREGATE",
  ARRAY = "ARRAY",

  // Enumerated values have both an integer (sint64Value) and a string representation
  ENUMERATED = "ENUMERATED",
  NONE = "NONE",
}

```

```

enum AcquisitionStatus {

    // OK!
    ACQUIRED = "ACQUIRED",

    // No value received so far
    NOT_RECEIVED = "NOT_RECEIVED",

    // Some value has been received but is invalid
    INVALID = "INVALID",

    // The parameter is coming from a packet which has not since updated although it should have been
    EXPIRED = "EXPIRED",
}

enum MonitoringResult {
    DISABLED = "DISABLED",
    IN_LIMITS = "IN_LIMITS",
    WATCH = "WATCH",
    WARNING = "WARNING",
    DISTRESS = "DISTRESS",
    CRITICAL = "CRITICAL",
    SEVERE = "SEVERE",
}

enum RangeCondition {
    LOW = "LOW",
    HIGH = "HIGH",
}

enum AlarmLevelType {
    NORMAL = "NORMAL",
    WATCH = "WATCH",
    WARNING = "WARNING",
    DISTRESS = "DISTRESS",
    CRITICAL = "CRITICAL",
    SEVERE = "SEVERE",
}

```

20.10 Batch Set Parameter Values

Batch set the value of multiple parameters

URI Template

```
POST /api/processors/{instance}/{processor}/parameters:batchSet
```

{instance}
Yamcs instance name.

{processor}
Processor name.

Request Body

```

interface BatchSetParameterValuesRequest {
    request: SetParameterValueRequest[];
}

```

Related Types

```
interface SetParameterValueRequest {
    id: NamedObjectId;
    value: Value;

    // The generation time of the value. If specified, must be a date
    // string in ISO 8601 format.
    generationTime: string; // RFC 3339 timestamp
}

// Used by external clients to identify an item in the Mission Database
// If namespace is set, then the name is that of an alias, rather than
// the qualified name.
interface NamedObjectId {
    name: string;
    namespace: string;
}

// Union type for storing a value
interface Value {
    type: Type;
    floatValue: number;
    doubleValue: number;
    sint32Value: number;
    uint32Value: number;
    binaryValue: string; // Base64
    stringValue: string;
    timestampValue: string; // String decimal
    uint64Value: string; // String decimal
    sint64Value: string; // String decimal
    booleanValue: boolean;
    aggregateValue: AggregateValue;
    arrayValue: Value[];
}

// An aggregate value is an ordered list of (member name, member value).
// Two arrays are used in order to be able to send just the values (since
// the names will not change)
interface AggregateValue {
    name: string[];
    value: Value[];
}

enum Type {
    FLOAT = "FLOAT",
    DOUBLE = "DOUBLE",
    UINT32 = "UINT32",
    SINT32 = "SINT32",
    BINARY = "BINARY",
    STRING = "STRING",
    TIMESTAMP = "TIMESTAMP",
    UINT64 = "UINT64",
    SINT64 = "SINT64",
    BOOLEAN = "BOOLEAN",
    AGGREGATE = "AGGREGATE",
    ARRAY = "ARRAY",

    // Enumerated values have both an integer (sint64Value) and a string representation
    ENUMERATED = "ENUMERATED",
    NONE = "NONE",
}
```

20.11 Subscribe TM Statistics

Receive TM statistics updates

WebSocket

This method requires to upgrade an HTTP connection to WebSocket. See details on [how Yamcs uses WebSocket¹⁵](#).

Use the message type `tmstats`.

Input Type

```
interface SubscribeTMStatisticsRequest {
    instance: string;
    processor: string;
}
```

Output Type

```
interface Statistics {

    // Yamcs instance name.
    instance: string;

    // Processor name.
    processor: string;
    tmstats: TmStatistics[];
    lastUpdated: string; // RFC 3339 timestamp
}
```

Related Types

```
interface TmStatistics {

    // Packet name.
    packetName: string;
    qualifiedName: string;
    receivedPackets: string; // String decimal
    subscribedParameterCount: number;
    lastReceived: string; // RFC 3339 timestamp
    lastPacketTime: string; // RFC 3339 timestamp
    packetRate: string; // String decimal
    dataRate: string; // String decimal
}
```

20.12 Subscribe Parameters

Receive parameter updates

The input message can be sent multiple types, allowing to alter a subscription with the `action` field.

WebSocket

This method requires to upgrade an HTTP connection to WebSocket. See details on [how Yamcs uses WebSocket¹⁶](#).

Use the message type `parameters`.

¹⁵ <https://docs.yamcs.org/yamcs-http-api/websocket>

¹⁶ <https://docs.yamcs.org/yamcs-http-api/websocket>

This method supports client-streaming. The reply on the first message includes the call identifier assigned by Yamcs. Ensure to specify this call identifier on subsequent messages, or Yamcs will assume that you are making a new unrelated call.

Input Type

```
interface SubscribeParametersRequest {  
  
    // Yamcs instance name.  
    instance: string;  
  
    // Processor name.  
    processor: string;  
    id: NamedObjectId[];  
  
    // Send an error message if any parameter is invalid.  
    // Default: true  
    abortOnInvalid: boolean;  
  
    // Send parameter updates when parameters expire.  
    // The update will have the same value and timestamp like  
    // the preceding update, but with acquisition status set to  
    // EXPIRED (instead of ACQUIRED)  
    // Default: false  
    updateOnExpiration: boolean;  
  
    // If available, send immediately the last cached value  
    // of each subscribed parameter.  
    // Default: true  
    sendFromCache: boolean;  
  
    // How to interpret the submitted parameter ids. Default  
    // is to replace an existing subscription with the newly  
    // submitted list.  
    action: Action;  
}
```

Output Type

```
interface SubscribeParametersData {  
  
    // mapping between numeric and subscribed id  
    mapping: {[key: number]: NamedObjectId};  
    invalid: NamedObjectId[];  
    values: ParameterValue[];  
}
```

Related Types

```
// Used by external clients to identify an item in the Mission Database  
// If namespace is set, then the name is that of an alias, rather than  
// the qualified name.  
interface NamedObjectId {  
    name: string;  
    namespace: string;  
}  
  
interface ParameterValue {  
  
    // Parameter identifier  
    id: NamedObjectId;  
  
    // Raw value (uncalibrated)
```

(continues on next page)

(continued from previous page)

```
rawValue: Value;

// Engineering value (calibrated)
engValue: Value;

// Time of Yamcs reception
acquisitionTime: string; // RFC 3339 timestamp

// Time of generation (~ packet time)
generationTime: string; // RFC 3339 timestamp
acquisitionStatus: AcquisitionStatus;

// Deprecated: this field was originally introduced for compatibility
// with Airbus CGS/CD-MCS system. It was redundant, because when false,
// the acquisitionStatus is also set to INVALID.
processingStatus: boolean;
monitoringResult: MonitoringResult;
rangeCondition: RangeCondition;

// Context-dependent ranges
alarmRange: AlarmRange[];

// How long (in milliseconds) this parameter value is valid
// Note that there is an option when subscribing to parameters to get
// updated when the parameter values expire.
expireMillis: string; // String decimal

// When transferring parameters over WebSocket, this value might be used
// instead of the id above in order to reduce the bandwidth.
// Note that the id <-> numericId assignment is only valid in the context
// of a single WebSocket call.
numericId: number;
}

// Union type for storing a value
interface Value {
  type: Type;
  floatValue: number;
  doubleValue: number;
  sint32Value: number;
  uint32Value: number;
  binaryValue: string; // Base64
  stringValue: string;
  timestampValue: string; // String decimal
  uint64Value: string; // String decimal
  sint64Value: string; // String decimal
  booleanValue: boolean;
  aggregateValue: AggregateValue;
  arrayValue: Value[];
}

// An aggregate value is an ordered list of (member name, member value).
// Two arrays are used in order to be able to send just the values (since
// the names will not change)
interface AggregateValue {
  name: string[];
  value: Value[];
}

interface AlarmRange {
  level: AlarmLevelType;
  minInclusive: number;
  maxInclusive: number;
  minExclusive: number;
  maxExclusive: number;
}

enum Action {
  REPLACE = "REPLACE",
  ADD = "ADD",
  REMOVE = "REMOVE",
}
```

(continues on next page)

```

}

enum Type {
    FLOAT = "FLOAT",
    DOUBLE = "DOUBLE",
    UINT32 = "UINT32",
    SINT32 = "SINT32",
    BINARY = "BINARY",
    STRING = "STRING",
    TIMESTAMP = "TIMESTAMP",
    UINT64 = "UINT64",
    SINT64 = "SINT64",
    BOOLEAN = "BOOLEAN",
    AGGREGATE = "AGGREGATE",
    ARRAY = "ARRAY",

    // Enumerated values have both an integer (sint64Value) and a string representation
    ENUMERATED = "ENUMERATED",
    NONE = "NONE",
}

enum AcquisitionStatus {

    // OK!
    ACQUIRED = "ACQUIRED",

    // No value received so far
    NOT_RECEIVED = "NOT_RECEIVED",

    // Some value has been received but is invalid
    INVALID = "INVALID",

    // The parameter is coming from a packet which has not since updated although it should have been
    EXPIRED = "EXPIRED",
}

enum MonitoringResult {
    DISABLED = "DISABLED",
    IN_LIMITS = "IN_LIMITS",
    WATCH = "WATCH",
    WARNING = "WARNING",
    DISTRESS = "DISTRESS",
    CRITICAL = "CRITICAL",
    SEVERE = "SEVERE",
}

enum RangeCondition {
    LOW = "LOW",
    HIGH = "HIGH",
}

enum AlarmLevelType {
    NORMAL = "NORMAL",
    WATCH = "WATCH",
    WARNING = "WARNING",
    DISTRESS = "DISTRESS",
    CRITICAL = "CRITICAL",
    SEVERE = "SEVERE",
}

```

20.13 Subscribe Processors

Receive processor updates

WebSocket

This method requires to upgrade an HTTP connection to WebSocket. See details on [how Yamcs uses WebSocket](#)¹⁷.

Use the message type processors.

Input Type

```
interface SubscribeProcessorsRequest {  
  
    // Yamcs instance name.  
    instance: string;  
  
    // Processor name.  
    processor: string;  
}
```

Output Type

```
interface ProcessorInfo {  
  
    // Yamcs instance name.  
    instance: string;  
  
    // Processor name.  
    name: string;  
    type: string;  
    spec: string;  
    creator: string;  
    hasAlarms: boolean;  
    hasCommanding: boolean;  
    state: ServiceState;  
    replayRequest: ReplayRequest;  
    replayState: ReplayState;  
    services: ServiceInfo[];  
    persistent: boolean;  
    time: string; // RFC 3339 timestamp  
    replay: boolean;  
    checkCommandClearance: boolean;  
}
```

Related Types

```
//used to replay (concurrently) TM packets, parameters and events  
interface ReplayRequest {  
  
    // **Required.** The time at which the replay should start.  
    start: string; // RFC 3339 timestamp  
  
    // The time at which the replay should stop.  
    // If unspecified, the replay will keep going as long as there is remaining data.  
    stop: string; // RFC 3339 timestamp  
  
    //what should happen at the end of the replay  
    endAction: EndAction;  
  
    //how fast the replay should go  
    speed: ReplaySpeed;  
  
    // Reverse the direction of the replay
```

(continues on next page)

¹⁷ <https://docs.yamcs.org/yamcs-http-api/websocket>

```

reverse: boolean;
parameterRequest: ParameterReplayRequest;

// By default all Packets, Events, CommandHistory are part of the replay
// Unless one or more of the below requests are specified.
packetRequest: PacketReplayRequest;
eventRequest: EventReplayRequest;
commandHistoryRequest: CommandHistoryReplayRequest;
ppRequest: PpReplayRequest;
}

interface ReplaySpeed {
    type: ReplaySpeedType;
    param: number;
}

interface ParameterReplayRequest {
    nameFilter: NamedObjectId[];
    sendRaw: boolean;
    performMonitoring: boolean;
}

// Used by external clients to identify an item in the Mission Database
// If namespace is set, then the name is that of an alias, rather than
// the qualified name.
interface NamedObjectId {
    name: string;
    namespace: string;
}

interface PacketReplayRequest {

    // No filter, means all packets for which privileges exist, are sent
    nameFilter: NamedObjectId[];
}

interface EventReplayRequest {
}

interface CommandHistoryReplayRequest {

    // No filter, means all command history entries are sent
    nameFilter: NamedObjectId[];
}

//Request to replay parameters - they can be filtered by the parameter group
interface PpReplayRequest {

    // No filter, means all pp groups are sent
    groupNameFilter: string[];

    // exclude the parameters from these groups
    // this takes precedence over the filter above (i.e. if a group is part of both, it will be excluded)
    groupNameExclude: string[];
}

interface ServiceInfo {
    instance: string;
    name: string;
    state: ServiceState;
    className: string;
    processor: string;
}

enum ServiceState {
    NEW = "NEW",
    STARTING = "STARTING",
    RUNNING = "RUNNING",
    STOPPING = "STOPPING",
    TERMINATED = "TERMINATED",
    FAILED = "FAILED",
}

```

```

}

enum EndAction {
    LOOP = "LOOP",
    QUIT = "QUIT",
    STOP = "STOP",
}

enum ReplaySpeedType {
    AFAP = "AFAP",
    FIXED_DELAY = "FIXED_DELAY",
    REALTIME = "REALTIME",
    STEP_BY_STEP = "STEP_BY_STEP",
}

enum ReplayState {

    // just at the beginning or when the replay request (start, stop or packet selection) changes
    INITIALIZATION = "INITIALIZATION",
    RUNNING = "RUNNING",

    // The replay has reached the end with the endaction stop
    STOPPED = "STOPPED",

    // The replay stopped due to an error.
    ERROR = "ERROR",
    PAUSED = "PAUSED",

    // The replay is finished and closed
    CLOSED = "CLOSED",
}

enum ServiceState {
    NEW = "NEW",
    STARTING = "STARTING",
    RUNNING = "RUNNING",
    STOPPING = "STOPPING",
    TERMINATED = "TERMINATED",
    FAILED = "FAILED",
}

```

20.14 Get Algorithm Status

Get the algorithm status

URI Template

```
GET /api/processors/{instance}/{processor}/algorithms/{name*}/status
```

{instance}
Yamcs instance name.

{processor}
Processor name.

{name*}
Algorithm name.

Response Type

```

interface AlgorithmStatus {

    //true if the algorithm is active
    active: boolean;

    //true if the tracing has been enabled
    traceEnabled: boolean;

    // how many times the algorithm ran (successfully or with error)
    runCount: number;

    // when the algorithm was last run
    lastRun: string; // RFC 3339 timestamp

    // how many times the algorithm ran with errors
    errorCount: number;

    // if the algorithm produced an error,
    // the fields below contain the error message and the time when the error was raised
    errorMessage: string;
    errorTime: string; // RFC 3339 timestamp

    //total execution time in nanoseconds
    execTimeNs: string; // String decimal
}

```

20.15 Subscribe Algorithm Status

Receive algorithm status updates

WebSocket

This method requires to upgrade an HTTP connection to WebSocket. See details on [how Yamcs uses WebSocket¹⁸](#).

Use the message type algorithm-status.

Input Type

```

interface SubscribeAlgorithmStatusRequest {

    // Yamcs instance name.
    instance: string;

    // Processor name.
    processor: string;

    // Algorithm name.
    name: string;
}

```

Output Type

```

interface AlgorithmStatus {

    //true if the algorithm is active
    active: boolean;

```

(continues on next page)

¹⁸ <https://docs.yamcs.org/yamcs-http-api/websocket>

(continued from previous page)

```
//true if the tracing has been enabled
traceEnabled: boolean;

// how many times the algorithm ran (successfully or with error)
runCount: number;

// when the algorithm was last run
lastRun: string; // RFC 3339 timestamp

// how many times the algorithm ran with errors
errorCount: number;

// if the algorithm produced an error,
// the fields below contain the error message and the time when the error was raised
errorMessage: string;
errorTime: string; // RFC 3339 timestamp

//total execution time in nanoseconds
execTimeNs: string; // String decimal
}
```

20.16 Get Algorithm Trace

Get the algorithm trace

URI Template

```
GET /api/processors/{instance}/{processor}/algorithms/{name*}/trace
```

{instance}
Yamcs instance name.

{processor}
Processor name.

{name*}
Algorithm name.

Response Type

```
interface AlgorithmTrace {
  runs: Run[];
  logs: Log[];
}
```

Related Types

```
interface Run {
  time: string; // RFC 3339 timestamp
  inputs: ParameterValue[];
  outputs: ParameterValue[];
  returnValue: string;
  error: string;
}

interface ParameterValue {
  // Parameter identifier
```

(continues on next page)

```

id: NamedObjectId;

// Raw value (uncalibrated)
rawValue: Value;

// Engineering value (calibrated)
engValue: Value;

// Time of Yamcs reception
acquisitionTime: string; // RFC 3339 timestamp

// Time of generation (~ packet time)
generationTime: string; // RFC 3339 timestamp
acquisitionStatus: AcquisitionStatus;

// Deprecated: this field was originally introduced for compatibility
// with Airbus CGS/CD-MCS system. It was redundant, because when false,
// the acquisitionStatus is also set to INVALID.
processingStatus: boolean;
monitoringResult: MonitoringResult;
rangeCondition: RangeCondition;

// Context-dependent ranges
alarmRange: AlarmRange[];

// How long (in milliseconds) this parameter value is valid
// Note that there is an option when subscribing to parameters to get
// updated when the parameter values expire.
expireMillis: string; // String decimal

// When transferring parameters over WebSocket, this value might be used
// instead of the id above in order to reduce the bandwidth.
// Note that the id <-> numericId assignment is only valid in the context
// of a single WebSocket call.
numericId: number;
}

// Used by external clients to identify an item in the Mission Database
// If namespace is set, then the name is that of an alias, rather than
// the qualified name.
interface NamedObjectId {
    name: string;
    namespace: string;
}

// Union type for storing a value
interface Value {
    type: Type;
    floatValue: number;
    doubleValue: number;
    sint32Value: number;
    uint32Value: number;
    binaryValue: string; // Base64
    stringValue: string;
    timestampValue: string; // String decimal
    uint64Value: string; // String decimal
    sint64Value: string; // String decimal
    booleanValue: boolean;
    aggregateValue: AggregateValue;
    arrayValue: Value[];
}

// An aggregate value is an ordered list of (member name, member value).
// Two arrays are used in order to be able to send just the values (since
// the names will not change)
interface AggregateValue {
    name: string[];
    value: Value[];
}

interface AlarmRange {

```

```

    level: AlarmLevelType;
    minInclusive: number;
    maxInclusive: number;
    minExclusive: number;
    maxExclusive: number;
}

interface Log {
    time: string; // RFC 3339 timestamp
    msg: string;
}

enum Type {
    FLOAT = "FLOAT",
    DOUBLE = "DOUBLE",
    UINT32 = "UINT32",
    SINT32 = "SINT32",
    BINARY = "BINARY",
    STRING = "STRING",
    TIMESTAMP = "TIMESTAMP",
    UINT64 = "UINT64",
    SINT64 = "SINT64",
    BOOLEAN = "BOOLEAN",
    AGGREGATE = "AGGREGATE",
    ARRAY = "ARRAY",

    // Enumerated values have both an integer (sint64Value) and a string representation
    ENUMERATED = "ENUMERATED",
    NONE = "NONE",
}

enum AcquisitionStatus {

    // OK!
    ACQUIRED = "ACQUIRED",

    // No value received so far
    NOT_RECEIVED = "NOT_RECEIVED",

    // Some value has been received but is invalid
    INVALID = "INVALID",

    // The parameter is coming from a packet which has not since updated although it should have been
    EXPIRED = "EXPIRED",
}

enum MonitoringResult {
    DISABLED = "DISABLED",
    IN_LIMITS = "IN_LIMITS",
    WATCH = "WATCH",
    WARNING = "WARNING",
    DISTRESS = "DISTRESS",
    CRITICAL = "CRITICAL",
    SEVERE = "SEVERE",
}

enum RangeCondition {
    LOW = "LOW",
    HIGH = "HIGH",
}

enum AlarmLevelType {
    NORMAL = "NORMAL",
    WATCH = "WATCH",
    WARNING = "WARNING",
    DISTRESS = "DISTRESS",
    CRITICAL = "CRITICAL",
    SEVERE = "SEVERE",
}

```

20.17 Edit Algorithm Trace

Enable/disable algorithm tracing

URI Template

```
PATCH /api/processors/{instance}/{processor}/algorithms/{name*}/trace
```

{instance}

Yamcs instance name.

{processor}

Processor name.

{name*}

Algorithm name.

Request Body

```
interface EditAlgorithmTraceRequest {  
    // Trace state: either ``enabled`` or ``disabled``.  
    state: string;  
}
```


21. Queue

21.1 List Queues

List command queues

URI Template

```
GET /api/processors/{instance}/{processor}/queues
```

{instance}
Yamcs instance nameee.

{processor}
Processor name.

Response Type

```
interface ListQueuesResponse {  
  queues: CommandQueueInfo[];  
}
```

Related Types

```
interface CommandQueueInfo {  
  instance: string;  
  processorName: string;  
  name: string;  
  state: QueueState;  
  nbSentCommands: number;  
  nbRejectedCommands: number;  
  stateExpirationTimeS: number;  
  entry: CommandQueueEntry[];  
  order: number;  
  users: string[];  
  groups: string[];  
  minLevel: SignificanceLevelType;  
}  
  
//One entry (command) in the command queue  
interface CommandQueueEntry {  
  instance: string;  
  processorName: string;  
  queueName: string;  
  id: string;  
  origin: string;  
  sequenceNumber: number;  
  commandName: string;
```

(continues on next page)

```

// Deprecated. If you require a string representation of this
// command, you can build it based on the fields ``commandName``
// and ``assignments``.
source: string;
assignments: CommandAssignment[];
binary: string; // Base64
username: string;

// Deprecated. This used to be used for manual accept/reject
// of a single queued command, however this can now be done
// using the ``id`` field instead.
uuid: string;
comment: string;
generationTime: string; // RFC 3339 timestamp

// If true, the command has been accepted and is due for release
// as soon as transmission constraints are satisfied.
pendingTransmissionConstraints: boolean;
}

interface CommandAssignment {
  name: string;
  value: Value;
  userInput: boolean;
}

// Union type for storing a value
interface Value {
  type: Type;
  floatValue: number;
  doubleValue: number;
  sint32Value: number;
  uint32Value: number;
  binaryValue: string; // Base64
  stringValue: string;
  timestampValue: string; // String decimal
  uint64Value: string; // String decimal
  sint64Value: string; // String decimal
  booleanValue: boolean;
  aggregateValue: AggregateValue;
  arrayValue: Value[];
}

// An aggregate value is an ordered list of (member name, member value).
// Two arrays are used in order to be able to send just the values (since
// the names will not change)
interface AggregateValue {
  name: string[];
  value: Value[];
}

enum QueueState {
  BLOCKED = "BLOCKED",
  DISABLED = "DISABLED",
  ENABLED = "ENABLED",
}

enum Type {
  FLOAT = "FLOAT",
  DOUBLE = "DOUBLE",
  UINT32 = "UINT32",
  SINT32 = "SINT32",
  BINARY = "BINARY",
  STRING = "STRING",
  TIMESTAMP = "TIMESTAMP",
  UINT64 = "UINT64",
  SINT64 = "SINT64",
  BOOLEAN = "BOOLEAN",
  AGGREGATE = "AGGREGATE",
  ARRAY = "ARRAY",
}

```

(continued from previous page)

```
// Enumerated values have both an integer (sint64Value) and a string representation
ENUMERATED = "ENUMERATED",
NONE = "NONE",
}

enum SignificanceLevelType {
    NONE = "NONE",
    WATCH = "WATCH",
    WARNING = "WARNING",
    DISTRESS = "DISTRESS",
    CRITICAL = "CRITICAL",
    SEVERE = "SEVERE",
}
```

21.2 Get Queue

Get a command queue

URI Template

```
GET /api/processors/{instance}/{processor}/queues/{queue}
```

{instance}
Yamcs instance name.

{processor}
Processor name.

{queue}
Queue name.

Response Type

```
interface CommandQueueInfo {
    instance: string;
    processorName: string;
    name: string;
    state: QueueState;
    nbSentCommands: number;
    nbRejectedCommands: number;
    stateExpirationTimeS: number;
    entry: CommandQueueEntry[];
    order: number;
    users: string[];
    groups: string[];
    minLevel: SignificanceLevelType;
}
```

Related Types

```
//One entry (command) in the command queue
interface CommandQueueEntry {
    instance: string;
    processorName: string;
    queueName: string;
    id: string;
    origin: string;
    sequenceNumber: number;
```

(continues on next page)

(continued from previous page)

```
commandName: string;

// Deprecated. If you require a string representation of this
// command, you can build it based on the fields ``commandName``
// and ``assignments``.
source: string;
assignments: CommandAssignment[];
binary: string; // Base64
username: string;

// Deprecated. This used to be used for manual accept/reject
// of a single queued command, however this can now be done
// using the ``id`` field instead.
uuid: string;
comment: string;
generationTime: string; // RFC 3339 timestamp

// If true, the command has been accepted and is due for release
// as soon as transmission constraints are satisfied.
pendingTransmissionConstraints: boolean;
}

interface CommandAssignment {
  name: string;
  value: Value;
  userInput: boolean;
}

// Union type for storing a value
interface Value {
  type: Type;
  floatValue: number;
  doubleValue: number;
  sint32Value: number;
  uint32Value: number;
  binaryValue: string; // Base64
  stringValue: string;
  timestampValue: string; // String decimal
  uint64Value: string; // String decimal
  sint64Value: string; // String decimal
  booleanValue: boolean;
  aggregateValue: AggregateValue;
  arrayValue: Value[];
}

// An aggregate value is an ordered list of (member name, member value).
// Two arrays are used in order to be able to send just the values (since
// the names will not change)
interface AggregateValue {
  name: string[];
  value: Value[];
}

enum QueueState {
  BLOCKED = "BLOCKED",
  DISABLED = "DISABLED",
  ENABLED = "ENABLED",
}

enum Type {
  FLOAT = "FLOAT",
  DOUBLE = "DOUBLE",
  UINT32 = "UINT32",
  SINT32 = "SINT32",
  BINARY = "BINARY",
  STRING = "STRING",
  TIMESTAMP = "TIMESTAMP",
  UINT64 = "UINT64",
  SINT64 = "SINT64",
  BOOLEAN = "BOOLEAN",
  AGGREGATE = "AGGREGATE",
}
```

(continues on next page)

(continued from previous page)

```
ARRAY = "ARRAY",

// Enumerated values have both an integer (sint64Value) and a string representation
ENUMERATED = "ENUMERATED",
NONE = "NONE",
}

enum SignificanceLevelType {
    NONE = "NONE",
    WATCH = "WATCH",
    WARNING = "WARNING",
    DISTRESS = "DISTRESS",
    CRITICAL = "CRITICAL",
    SEVERE = "SEVERE",
}
```

21.3 Update Queue

Update a command queue

URI Template

```
PATCH /api/processors/{instance}/{processor}/queues/{queue}
```

{instance}
Yamcs instance name.

{processor}
Processor name.

{queue}
Queue name.

Request Body

```
interface EditQueueRequest {

    // The state of the queue. Either ``enabled``, ``disabled`` or ``blocked``.
    state: string;
}
```

Response Type

```
interface CommandQueueInfo {
    instance: string;
    processorName: string;
    name: string;
    state: QueueState;
    nbSentCommands: number;
    nbRejectedCommands: number;
    stateExpirationTimeS: number;
    entry: CommandQueueEntry[];
    order: number;
    users: string[];
    groups: string[];
    minLevel: SignificanceLevelType;
}
```

Related Types

```
//One entry (command) in the command queue
interface CommandQueueEntry {
    instance: string;
    processorName: string;
    queueName: string;
    id: string;
    origin: string;
    sequenceNumber: number;
    commandName: string;

    // Deprecated. If you require a string representation of this
    // command, you can build it based on the fields ``commandName``
    // and ``assignments``.
    source: string;
    assignments: CommandAssignment[];
    binary: string; // Base64
    username: string;

    // Deprecated. This used to be used for manual accept/reject
    // of a single queued command, however this can now be done
    // using the ``id`` field instead.
    uuid: string;
    comment: string;
    generationTime: string; // RFC 3339 timestamp

    // If true, the command has been accepted and is due for release
    // as soon as transmission constraints are satisfied.
    pendingTransmissionConstraints: boolean;
}

interface CommandAssignment {
    name: string;
    value: Value;
    userInput: boolean;
}

// Union type for storing a value
interface Value {
    type: Type;
    floatValue: number;
    doubleValue: number;
    sint32Value: number;
    uint32Value: number;
    binaryValue: string; // Base64
    stringValue: string;
    timestampValue: string; // String decimal
    uint64Value: string; // String decimal
    sint64Value: string; // String decimal
    booleanValue: boolean;
    aggregateValue: AggregateValue;
    arrayValue: Value[];
}

// An aggregate value is an ordered list of (member name, member value).
// Two arrays are used in order to be able to send just the values (since
// the names will not change)
interface AggregateValue {
    name: string[];
    value: Value[];
}

enum QueueState {
    BLOCKED = "BLOCKED",
    DISABLED = "DISABLED",
    ENABLED = "ENABLED",
}

enum Type {
    FLOAT = "FLOAT",
```

(continues on next page)

```

DOUBLE = "DOUBLE",
UINT32 = "UINT32",
SINT32 = "SINT32",
BINARY = "BINARY",
STRING = "STRING",
TIMESTAMP = "TIMESTAMP",
UINT64 = "UINT64",
SINT64 = "SINT64",
BOOLEAN = "BOOLEAN",
AGGREGATE = "AGGREGATE",
ARRAY = "ARRAY",

// Enumerated values have both an integer (sint64Value) and a string representation
ENUMERATED = "ENUMERATED",
NONE = "NONE",
}

enum SignificanceLevelType {
    NONE = "NONE",
    WATCH = "WATCH",
    WARNING = "WARNING",
    DISTRESS = "DISTRESS",
    CRITICAL = "CRITICAL",
    SEVERE = "SEVERE",
}

```

21.4 Enable Queue

Enable a command queue

URI Template

```
POST /api/processors/{instance}/{processor}/queues/{queue}:enable
```

{instance}
Yamcs instance name.

{processor}
Processor name.

{queue}
Queue name.

Response Type

```

interface CommandQueueInfo {
    instance: string;
    processorName: string;
    name: string;
    state: QueueState;
    nbSentCommands: number;
    nbRejectedCommands: number;
    stateExpirationTimeS: number;
    entry: CommandQueueEntry[];
    order: number;
    users: string[];
    groups: string[];
    minLevel: SignificanceLevelType;
}

```

Related Types

```
//One entry (command) in the command queue
interface CommandQueueEntry {
    instance: string;
    processorName: string;
    queueName: string;
    id: string;
    origin: string;
    sequenceNumber: number;
    commandName: string;

    // Deprecated. If you require a string representation of this
    // command, you can build it based on the fields ``commandName``
    // and ``assignments``.
    source: string;
    assignments: CommandAssignment[];
    binary: string; // Base64
    username: string;

    // Deprecated. This used to be used for manual accept/reject
    // of a single queued command, however this can now be done
    // using the ``id`` field instead.
    uuid: string;
    comment: string;
    generationTime: string; // RFC 3339 timestamp

    // If true, the command has been accepted and is due for release
    // as soon as transmission constraints are satisfied.
    pendingTransmissionConstraints: boolean;
}

interface CommandAssignment {
    name: string;
    value: Value;
    userInput: boolean;
}

// Union type for storing a value
interface Value {
    type: Type;
    floatValue: number;
    doubleValue: number;
    sint32Value: number;
    uint32Value: number;
    binaryValue: string; // Base64
    stringValue: string;
    timestampValue: string; // String decimal
    uint64Value: string; // String decimal
    sint64Value: string; // String decimal
    booleanValue: boolean;
    aggregateValue: AggregateValue;
    arrayValue: Value[];
}

// An aggregate value is an ordered list of (member name, member value).
// Two arrays are used in order to be able to send just the values (since
// the names will not change)
interface AggregateValue {
    name: string[];
    value: Value[];
}

enum QueueState {
    BLOCKED = "BLOCKED",
    DISABLED = "DISABLED",
    ENABLED = "ENABLED",
}

enum Type {
    FLOAT = "FLOAT",
```

(continues on next page)

(continued from previous page)

```
DOUBLE = "DOUBLE",
UINT32 = "UINT32",
SINT32 = "SINT32",
BINARY = "BINARY",
STRING = "STRING",
TIMESTAMP = "TIMESTAMP",
UINT64 = "UINT64",
SINT64 = "SINT64",
BOOLEAN = "BOOLEAN",
AGGREGATE = "AGGREGATE",
ARRAY = "ARRAY",

// Enumerated values have both an integer (sint64Value) and a string representation
ENUMERATED = "ENUMERATED",
NONE = "NONE",
}

enum SignificanceLevelType {
    NONE = "NONE",
    WATCH = "WATCH",
    WARNING = "WARNING",
    DISTRESS = "DISTRESS",
    CRITICAL = "CRITICAL",
    SEVERE = "SEVERE",
}
```

21.5 Disable Queue

Disable a command queue

URI Template

```
POST /api/processors/{instance}/{processor}/queues/{queue}:disable
```

{instance}
Yamcs instance name.

{processor}
Processor name.

{queue}
Queue name.

Response Type

```
interface CommandQueueInfo {
    instance: string;
    processorName: string;
    name: string;
    state: QueueState;
    nbSentCommands: number;
    nbRejectedCommands: number;
    stateExpirationTimeS: number;
    entry: CommandQueueEntry[];
    order: number;
    users: string[];
    groups: string[];
    minLevel: SignificanceLevelType;
}
```

Related Types

```
//One entry (command) in the command queue
interface CommandQueueEntry {
    instance: string;
    processorName: string;
    queueName: string;
    id: string;
    origin: string;
    sequenceNumber: number;
    commandName: string;

    // Deprecated. If you require a string representation of this
    // command, you can build it based on the fields ``commandName``
    // and ``assignments``.
    source: string;
    assignments: CommandAssignment[];
    binary: string; // Base64
    username: string;

    // Deprecated. This used to be used for manual accept/reject
    // of a single queued command, however this can now be done
    // using the ``id`` field instead.
    uuid: string;
    comment: string;
    generationTime: string; // RFC 3339 timestamp

    // If true, the command has been accepted and is due for release
    // as soon as transmission constraints are satisfied.
    pendingTransmissionConstraints: boolean;
}

interface CommandAssignment {
    name: string;
    value: Value;
    userInput: boolean;
}

// Union type for storing a value
interface Value {
    type: Type;
    floatValue: number;
    doubleValue: number;
    sint32Value: number;
    uint32Value: number;
    binaryValue: string; // Base64
    stringValue: string;
    timestampValue: string; // String decimal
    uint64Value: string; // String decimal
    sint64Value: string; // String decimal
    booleanValue: boolean;
    aggregateValue: AggregateValue;
    arrayValue: Value[];
}

// An aggregate value is an ordered list of (member name, member value).
// Two arrays are used in order to be able to send just the values (since
// the names will not change)
interface AggregateValue {
    name: string[];
    value: Value[];
}

enum QueueState {
    BLOCKED = "BLOCKED",
    DISABLED = "DISABLED",
    ENABLED = "ENABLED",
}

enum Type {
    FLOAT = "FLOAT",
```

(continues on next page)

```

DOUBLE = "DOUBLE",
UINT32 = "UINT32",
SINT32 = "SINT32",
BINARY = "BINARY",
STRING = "STRING",
TIMESTAMP = "TIMESTAMP",
UINT64 = "UINT64",
SINT64 = "SINT64",
BOOLEAN = "BOOLEAN",
AGGREGATE = "AGGREGATE",
ARRAY = "ARRAY",

// Enumerated values have both an integer (sint64Value) and a string representation
ENUMERATED = "ENUMERATED",
NONE = "NONE",
}

enum SignificanceLevelType {
    NONE = "NONE",
    WATCH = "WATCH",
    WARNING = "WARNING",
    DISTRESS = "DISTRESS",
    CRITICAL = "CRITICAL",
    SEVERE = "SEVERE",
}

```

21.6 Block Queue

Block a command queue

URI Template

```
POST /api/processors/{instance}/{processor}/queues/{queue}:block
```

{instance}
Yamcs instance name.

{processor}
Processor name.

{queue}
Queue name.

Response Type

```

interface CommandQueueInfo {
    instance: string;
    processorName: string;
    name: string;
    state: QueueState;
    nbSentCommands: number;
    nbRejectedCommands: number;
    stateExpirationTimeS: number;
    entry: CommandQueueEntry[];
    order: number;
    users: string[];
    groups: string[];
    minLevel: SignificanceLevelType;
}

```

Related Types

```
//One entry (command) in the command queue
interface CommandQueueEntry {
    instance: string;
    processorName: string;
    queueName: string;
    id: string;
    origin: string;
    sequenceNumber: number;
    commandName: string;

    // Deprecated. If you require a string representation of this
    // command, you can build it based on the fields ``commandName``
    // and ``assignments``.
    source: string;
    assignments: CommandAssignment[];
    binary: string; // Base64
    username: string;

    // Deprecated. This used to be used for manual accept/reject
    // of a single queued command, however this can now be done
    // using the ``id`` field instead.
    uuid: string;
    comment: string;
    generationTime: string; // RFC 3339 timestamp

    // If true, the command has been accepted and is due for release
    // as soon as transmission constraints are satisfied.
    pendingTransmissionConstraints: boolean;
}

interface CommandAssignment {
    name: string;
    value: Value;
    userInput: boolean;
}

// Union type for storing a value
interface Value {
    type: Type;
    floatValue: number;
    doubleValue: number;
    sint32Value: number;
    uint32Value: number;
    binaryValue: string; // Base64
    stringValue: string;
    timestampValue: string; // String decimal
    uint64Value: string; // String decimal
    sint64Value: string; // String decimal
    booleanValue: boolean;
    aggregateValue: AggregateValue;
    arrayValue: Value[];
}

// An aggregate value is an ordered list of (member name, member value).
// Two arrays are used in order to be able to send just the values (since
// the names will not change)
interface AggregateValue {
    name: string[];
    value: Value[];
}

enum QueueState {
    BLOCKED = "BLOCKED",
    DISABLED = "DISABLED",
    ENABLED = "ENABLED",
}

enum Type {
    FLOAT = "FLOAT",
```

(continues on next page)

(continued from previous page)

```
DOUBLE = "DOUBLE",
UINT32 = "UINT32",
SINT32 = "SINT32",
BINARY = "BINARY",
STRING = "STRING",
TIMESTAMP = "TIMESTAMP",
UINT64 = "UINT64",
SINT64 = "SINT64",
BOOLEAN = "BOOLEAN",
AGGREGATE = "AGGREGATE",
ARRAY = "ARRAY",

// Enumerated values have both an integer (sint64Value) and a string representation
ENUMERATED = "ENUMERATED",
NONE = "NONE",
}

enum SignificanceLevelType {
    NONE = "NONE",
    WATCH = "WATCH",
    WARNING = "WARNING",
    DISTRESS = "DISTRESS",
    CRITICAL = "CRITICAL",
    SEVERE = "SEVERE",
}
```

21.7 Subscribe Queue Statistics

Receive updates on queue stats

WebSocket

This method requires to upgrade an HTTP connection to WebSocket. See details on [how Yamcs uses WebSocket](#)¹⁹.

Use the message type queue-stats.

Input Type

```
interface SubscribeQueueStatisticsRequest {

    // Yamcs instance name.
    instance: string;

    // Processor name.
    processor: string;
}
```

Output Type

```
interface CommandQueueInfo {
    instance: string;
    processorName: string;
    name: string;
    state: QueueState;
    nbSentCommands: number;
    nbRejectedCommands: number;
}
```

(continues on next page)

¹⁹ <https://docs.yamcs.org/yamcs-http-api/websocket>

```

stateExpirationTimeS: number;
entry: CommandQueueEntry[];
order: number;
users: string[];
groups: string[];
minLevel: SignificanceLevelType;
}

```

Related Types

```

//One entry (command) in the command queue
interface CommandQueueEntry {
  instance: string;
  processorName: string;
  queueName: string;
  id: string;
  origin: string;
  sequenceNumber: number;
  commandName: string;

  // Deprecated. If you require a string representation of this
  // command, you can build it based on the fields ``commandName``
  // and ``assignments``.
  source: string;
  assignments: CommandAssignment[];
  binary: string; // Base64
  username: string;

  // Deprecated. This used to be used for manual accept/reject
  // of a single queued command, however this can now be done
  // using the ``id`` field instead.
  uuid: string;
  comment: string;
  generationTime: string; // RFC 3339 timestamp

  // If true, the command has been accepted and is due for release
  // as soon as transmission constraints are satisfied.
  pendingTransmissionConstraints: boolean;
}

interface CommandAssignment {
  name: string;
  value: Value;
  userInput: boolean;
}

// Union type for storing a value
interface Value {
  type: Type;
  floatValue: number;
  doubleValue: number;
  sint32Value: number;
  uint32Value: number;
  binaryValue: string; // Base64
  stringValue: string;
  timestampValue: string; // String decimal
  uint64Value: string; // String decimal
  sint64Value: string; // String decimal
  booleanValue: boolean;
  aggregateValue: AggregateValue;
  arrayValue: Value[];
}

// An aggregate value is an ordered list of (member name, member value).
// Two arrays are used in order to be able to send just the values (since
// the names will not change)
interface AggregateValue {
  name: string[];

```

```

    value: Value[];
}

enum QueueState {
    BLOCKED = "BLOCKED",
    DISABLED = "DISABLED",
    ENABLED = "ENABLED",
}

enum Type {
    FLOAT = "FLOAT",
    DOUBLE = "DOUBLE",
    UINT32 = "UINT32",
    SINT32 = "SINT32",
    BINARY = "BINARY",
    STRING = "STRING",
    TIMESTAMP = "TIMESTAMP",
    UINT64 = "UINT64",
    SINT64 = "SINT64",
    BOOLEAN = "BOOLEAN",
    AGGREGATE = "AGGREGATE",
    ARRAY = "ARRAY",

    // Enumerated values have both an integer (sint64Value) and a string representation
    ENUMERATED = "ENUMERATED",
    NONE = "NONE",
}

enum SignificanceLevelType {
    NONE = "NONE",
    WATCH = "WATCH",
    WARNING = "WARNING",
    DISTRESS = "DISTRESS",
    CRITICAL = "CRITICAL",
    SEVERE = "SEVERE",
}

```

21.8 Subscribe Queue Events

Receive updates on queue events

WebSocket

This method requires to upgrade an HTTP connection to WebSocket. See details on [how Yamcs uses WebSocket](https://docs.yamcs.org/yamcs-http-api/websocket)²⁰.

Use the message type `queue-events`.

Input Type

```

interface SubscribeQueueEventsRequest {

    // Yamcs instance name.
    instance: string;

    // Processor name.
    processor: string;
}

```

²⁰ <https://docs.yamcs.org/yamcs-http-api/websocket>

Output Type

```
interface CommandQueueEvent {  
  type: Type;  
  data: CommandQueueEntry;  
}
```

Related Types

```
//One entry (command) in the command queue  
interface CommandQueueEntry {  
  instance: string;  
  processorName: string;  
  queueName: string;  
  id: string;  
  origin: string;  
  sequenceNumber: number;  
  commandName: string;  
  
  // Deprecated. If you require a string representation of this  
  // command, you can build it based on the fields ``commandName``  
  // and ``assignments``.  
  source: string;  
  assignments: CommandAssignment[];  
  binary: string; // Base64  
  username: string;  
  
  // Deprecated. This used to be used for manual accept/reject  
  // of a single queued command, however this can now be done  
  // using the ``id`` field instead.  
  uuid: string;  
  comment: string;  
  generationTime: string; // RFC 3339 timestamp  
  
  // If true, the command has been accepted and is due for release  
  // as soon as transmission constraints are satisfied.  
  pendingTransmissionConstraints: boolean;  
}  
  
interface CommandAssignment {  
  name: string;  
  value: Value;  
  userInput: boolean;  
}  
  
// Union type for storing a value  
interface Value {  
  type: Type;  
  floatValue: number;  
  doubleValue: number;  
  sint32Value: number;  
  uint32Value: number;  
  binaryValue: string; // Base64  
  stringValue: string;  
  timestampValue: string; // String decimal  
  uint64Value: string; // String decimal  
  sint64Value: string; // String decimal  
  booleanValue: boolean;  
  aggregateValue: AggregateValue;  
  arrayValue: Value[];  
}  
  
// An aggregate value is an ordered list of (member name, member value).  
// Two arrays are used in order to be able to send just the values (since  
// the names will not change)  
interface AggregateValue {  
  name: string[];  
  value: Value[];  
}
```

(continues on next page)

(continued from previous page)

```
enum Type {
    COMMAND_ADDED = "COMMAND_ADDED",
    COMMAND_REJECTED = "COMMAND_REJECTED",
    COMMAND_SENT = "COMMAND_SENT",
    COMMAND_UPDATED = "COMMAND_UPDATED",
}

enum Type {
    FLOAT = "FLOAT",
    DOUBLE = "DOUBLE",
    UINT32 = "UINT32",
    SINT32 = "SINT32",
    BINARY = "BINARY",
    STRING = "STRING",
    TIMESTAMP = "TIMESTAMP",
    UINT64 = "UINT64",
    SINT64 = "SINT64",
    BOOLEAN = "BOOLEAN",
    AGGREGATE = "AGGREGATE",
    ARRAY = "ARRAY",

    // Enumerated values have both an integer (sint64Value) and a string representation
    ENUMERATED = "ENUMERATED",
    NONE = "NONE",
}
```

21.9 List Queued Commands

List queued commands

URI Template

```
GET /api/processors/{instance}/{processor}/queues/{queue}/commands
```

{instance}
Yamcs instance name.

{processor}
Processor name.

{queue}
Queue name.

Response Type

```
interface ListQueuedCommandsResponse {
    commands: CommandQueueEntry[];
}
```

Related Types

```
//One entry (command) in the command queue
interface CommandQueueEntry {
    instance: string;
    processorName: string;
    queueName: string;
    id: string;
```

(continues on next page)

```

origin: string;
sequenceNumber: number;
commandName: string;

// Deprecated. If you require a string representation of this
// command, you can build it based on the fields ``commandName``
// and ``assignments``.
source: string;
assignments: CommandAssignment[];
binary: string; // Base64
username: string;

// Deprecated. This used to be used for manual accept/reject
// of a single queued command, however this can now be done
// using the ``id`` field instead.
uuid: string;
comment: string;
generationTime: string; // RFC 3339 timestamp

// If true, the command has been accepted and is due for release
// as soon as transmission constraints are satisfied.
pendingTransmissionConstraints: boolean;
}

interface CommandAssignment {
  name: string;
  value: Value;
  userInput: boolean;
}

// Union type for storing a value
interface Value {
  type: Type;
  floatValue: number;
  doubleValue: number;
  sint32Value: number;
  uint32Value: number;
  binaryValue: string; // Base64
  stringValue: string;
  timestampValue: string; // String decimal
  uint64Value: string; // String decimal
  sint64Value: string; // String decimal
  booleanValue: boolean;
  aggregateValue: AggregateValue;
  arrayValue: Value[];
}

// An aggregate value is an ordered list of (member name, member value).
// Two arrays are used in order to be able to send just the values (since
// the names will not change)
interface AggregateValue {
  name: string[];
  value: Value[];
}

enum Type {
  FLOAT = "FLOAT",
  DOUBLE = "DOUBLE",
  UINT32 = "UINT32",
  SINT32 = "SINT32",
  BINARY = "BINARY",
  STRING = "STRING",
  TIMESTAMP = "TIMESTAMP",
  UINT64 = "UINT64",
  SINT64 = "SINT64",
  BOOLEAN = "BOOLEAN",
  AGGREGATE = "AGGREGATE",
  ARRAY = "ARRAY",

  // Enumerated values have both an integer (sint64Value) and a string representation
  ENUMERATED = "ENUMERATED",

```

```
NONE = "NONE",
}
```

21.10 Update Queue Entry

Update a command queue entry

URI Template

```
PATCH /api/processors/{instance}/{processor}/queues/{queue}/entries/{uuid}
```

{instance}
Yamcs instance name.

{processor}
Processor name.

{queue}
Queue name.

{uuid}

Request Body

```
interface EditQueueEntryRequest {
    // The state of the entry. Either ``released`` or ``rejected``.
    state: string;
}
```

21.11 Accept Command

Accept a queued command

URI Template

```
POST /api/processors/{instance}/{processor}/queues/{queue}/commands/{command}:accept
```

{instance}
Yamcs instance name.

{processor}
Processor name.

{queue}
Queue name.

{command}
Command identifier.

21.12 Reject Command

Reject a queued command

URI Template

```
POST /api/processors/{instance}/{processor}/queues/{queue}/commands/{command}:reject
```

{instance}

Yamcs instance name.

{processor}

Processor name.

{queue}

Queue name.

{command}

Command identifier.

22. Replication

22.1 Get Replication Info

Get replication info

URI Template

```
GET /api/replication
```

Response Type

```
interface ReplicationInfo {  
    masters: ReplicationMasterInfo[];  
    slaves: ReplicationSlaveInfo[];  
}
```

Related Types

```
interface ReplicationMasterInfo {  
    instance: string;  
    streams: string[];  
    localAddress: string;  
    remoteAddress: string;  
    push: boolean;  
    pushTo: string;  
    localTx: string; // String decimal  
    nextTx: string; // String decimal  
}  
  
interface ReplicationSlaveInfo {  
    instance: string;  
    streams: string[];  
    localAddress: string;  
    remoteAddress: string;  
    push: boolean;  
    pullFrom: string;  
    tx: string; // String decimal  
}
```

22.2 Subscribe Replication Info

Receive replication updates

WebSocket

This method requires to upgrade an HTTP connection to WebSocket. See details on [how Yamcs uses WebSocket](#)²¹.

Use the message type replication-info.

Output Type

```
interface ReplicationInfo {  
    masters: ReplicationMasterInfo[];  
    slaves: ReplicationSlaveInfo[];  
}
```

Related Types

```
interface ReplicationMasterInfo {  
    instance: string;  
    streams: string[];  
    localAddress: string;  
    remoteAddress: string;  
    push: boolean;  
    pushTo: string;  
    localTx: string; // String decimal  
    nextTx: string; // String decimal  
}  
  
interface ReplicationSlaveInfo {  
    instance: string;  
    streams: string[];  
    localAddress: string;  
    remoteAddress: string;  
    push: boolean;  
    pullFrom: string;  
    tx: string; // String decimal  
}
```

²¹ <https://docs.yamcs.org/yamcs-http-api/websocket>

23. Rocks Db

23.1 List Tablespaces

List tablespaces

URI Template

```
GET /api/archive/rocksdb/tablespaces
```

Response Type

```
interface ListRocksDbTablespacesResponse {  
  tablespaces: RocksDbTablespaceInfo[];  
}
```

Related Types

```
interface RocksDbTablespaceInfo {  
  name: string;  
  dataDir: string;  
  databases: RocksDbDatabaseInfo[];  
}  
  
interface RocksDbDatabaseInfo {  
  tablespace: string;  
  dataDir: string;  
  dbPath: string;  
}
```

23.2 Backup Database

Backup database

URI Template

```
POST /api/archive/rocksdb/{tablespace}/{dbpath*}:backup
```

{tablespace}

{dbpath*}

23.3 List Databases

List databases

URI Template

```
GET /api/archive/rocksdb/databases
```

Response Type

```
interface ListRocksDbDatabasesResponse {  
  databases: RocksDbDatabaseInfo[];  
}
```

Related Types

```
interface RocksDbDatabaseInfo {  
  tablespace: string;  
  dataDir: string;  
  dbPath: string;  
}
```

23.4 Compact Database

Compact database

URI Template

```
POST /api/archive/rocksdb/{tablespace}/{dbpath**}:compact
```

{tablespace}

{dbpath**}

23.5 Describe Rocks Db

Get a text-dump with general RocksDB info

URI Template

```
GET /api/archive/rocksdb:describe
```


23.6 Describe Database

Get a text-dump describing a database

This operation can be used to debug the inner workings of RocksDB database. For example the property `rocksdb.estimate-table-readers-mem` will provide an estimation of how much memory is used by the index and filter cache of RocksDB (note that the memory used by RocksDB is outside the java heap space).

See also: <https://github.com/facebook/rocksdb/blob/master/include/rocksdb/db.h>

The response contains a dump of various rocksdb properties for each column family. The single value properties are presented in a “name: value” list. The multiline properties are preceded by a line including the property name between dashes.

URI Template

```
GET /api/archive/rocksdb/{tablespace}/{dbpath**}:describe
```

{tablespace}

{dbpath}**

24. Server

Handles incoming requests related to api routes

24.1 Get Server Info

Get general server info

URI Template

```
GET /api
```

Response Type

```
interface GetServerInfoResponse {  
  
    // Yamcs version derived on build time.  
    yamcsVersion: string;  
  
    // Yamcs SHA-1 revision identifier. Set on  
    // build time, but only if the git command  
    // was available.  
    revision: string;  
  
    // An identifier for this server. Used in  
    // system parameters.  
    serverId: string;  
  
    // A default instance for this Yamcs installation.  
    // This is a calculated suggestion. UI clients may ignore.  
    defaultYamcsInstance: string;  
  
    // Plugins loaded within this server instance  
    plugins: PluginInfo[];  
  
    // Additional options available to commands  
    commandOptions: CommandOptionInfo[];  
}
```

Related Types

```
interface PluginInfo {  
    name: string;  
    description: string;  
    version: string;  
    vendor: string;  
}
```

(continues on next page)

(continued from previous page)

```
interface CommandOptionInfo {
    id: string;
    verboseName: string;
    type: string;
    help: string;
}
```

24.2 List Routes

List routes

URI Template

```
GET /api/routes
```

Response Type

```
interface ListRoutesResponse {
    routes: RouteInfo[];
}
```

Related Types

```
interface RouteInfo {
    service: string;
    method: string;
    description: string;
    httpMethod: string;
    url: string;
    inputType: string;
    outputType: string;
    deprecated: boolean;
    requestCount: string; // String decimal
    errorCount: string; // String decimal
    logFormat: string;
}
```

24.3 List Topics

List topics

URI Template

```
GET /api/topics
```

Response Type

```
interface ListTopicsResponse {
    topics: TopicInfo[];
}
```

Related Types

```
interface TopicInfo {
  topic: string;
  service: string;
  method: string;
  description: string;
  inputType: string;
  outputType: string;
  deprecated: boolean;
}
```

24.4 List Threads

List threads

URI Template

```
GET /api/threads
```

Query Parameters

depth

Maximum depth of each thread's stacktrace. Default: no limit.

Response Type

```
interface ListThreadsResponse {
  threads: ThreadInfo[];
}
```

Related Types

```
interface ThreadInfo {
  id: string; // String decimal
  name: string;
  state: string;
  native: boolean;
  suspended: boolean;
  group: ThreadGroupInfo;
  trace: TraceElementInfo[];
}
```

```
interface ThreadGroupInfo {
  name: string;
  daemon: boolean;
  parent: ThreadGroupInfo;
}
```

```
interface TraceElementInfo {
  className: string;
  fileName: string;
  methodName: string;
  lineNumber: number;
}
```

24.5 Get Thread

Get info on a single thread

URI Template

```
GET /api/threads/{id}
```

{id}

Thread ID

Response Type

```
interface ThreadInfo {  
    id: string; // String decimal  
    name: string;  
    state: string;  
    native: boolean;  
    suspended: boolean;  
    group: ThreadGroupInfo;  
    trace: TraceElementInfo[];  
}
```

Related Types

```
interface ThreadGroupInfo {  
    name: string;  
    daemon: boolean;  
    parent: ThreadGroupInfo;  
}  
  
interface TraceElementInfo {  
    className: string;  
    fileName: string;  
    methodName: string;  
    lineNumber: number;  
}
```

24.6 Dump Threads

Get a text-dump with thread information

URI Template

```
GET /api/threads:dump
```

24.7 Get Http Traffic

Get HTTP traffic

URI Template

```
GET /api/http-traffic
```

Response Type

```
interface HttpTraffic {  
  readBytes: string; // String decimal  
  writtenBytes: string; // String decimal  
  readThroughput: string; // String decimal  
  writeThroughput: string; // String decimal  
  connections: ClientConnectionInfo[];  
}
```

Related Types

```
interface ClientConnectionInfo {  
  id: string;  
  open: boolean;  
  active: boolean;  
  writable: boolean;  
  remoteAddress: string;  
  readBytes: string; // String decimal  
  writtenBytes: string; // String decimal  
  readThroughput: string; // String decimal  
  writeThroughput: string; // String decimal  
  httpRequest: HttpRequestInfo;  
  username: string;  
}  
  
interface HttpRequestInfo {  
  protocol: string;  
  method: string;  
  uri: string;  
  keepAlive: boolean;  
  userAgent: string;  
}
```

25. Sessions

25.1 List Sessions

List sessions

URI Template

```
GET /api/sessions
```

Response Type

```
interface ListSessionsResponse {  
    sessions: SessionInfo[];  
}
```

Related Types

```
interface SessionInfo {  
  
    // Session identifier  
    id: string;  
    username: string;  
    ipAddress: string;  
    hostname: string;  
    startTime: string; // RFC 3339 timestamp  
    lastAccessTime: string; // RFC 3339 timestamp  
    expirationTime: string; // RFC 3339 timestamp  
    clients: string[];  
}
```

26. Stream Archive

26.1 List Parameter Groups

List parameter groups

URI Template

```
GET /api/archive/{instance}/parameter-groups
```

{instance}

Response Type

```
interface ParameterGroupInfo {  
    group: string[];  
}
```

26.2 List Parameter History

List parameter history

URI Template

```
GET /api/stream-archive/{instance}/parameters/{name*}
```

{instance}

Yamcs instance name.

{name*}

Parameter name.

Query Parameters

pos

The zero-based row number at which to start outputting results. Default: 0.

limit

The maximum number of returned records per page. Choose this value too high and you risk hitting the maximum response size limit enforced by the server. Default: 100.

norepeat

Whether to filter out consecutive identical values. Default no.

start

Filter the lower bound of the parameter's generation time. Specify a date string in ISO 8601 format.

stop

Filter the upper bound of the parameter's generation time. Specify a date string in ISO 8601 format.

order

The order of the returned results. Can be either asc or desc. Default: desc.

norealtime

Disable loading of parameters from the parameter cache. Default: false.

processor

The name of the processor from which to use the parameter cache. Default: realtime.

source

Specifies how to retrieve the parameters. Either ParameterArchive or replay. If replay is specified, a replay processor will be created and data will be processed with the active Mission Database. Note that this is much slower than receiving data from the ParameterArchive.

Default: ParameterArchive.

next

Continuation token returned by a previous page response.

Response Type

```
interface ListParameterHistoryResponse {
    parameter: ParameterValue[];

    // Token indicating the response is only partial. More results can then
    // be obtained by performing the same request (including all original
    // query parameters) and setting the ``next`` parameter to this token.
    continuationToken: string;
}
```

Related Types

```
interface ParameterValue {

    // Parameter identifier
    id: NamedObjectId;

    // Raw value (uncalibrated)
    rawValue: Value;

    // Engineering value (calibrated)
```

(continues on next page)

```

engValue: Value;

// Time of Yamcs reception
acquisitionTime: string; // RFC 3339 timestamp

// Time of generation (~ packet time)
generationTime: string; // RFC 3339 timestamp
acquisitionStatus: AcquisitionStatus;

// Deprecated: this field was originally introduced for compatibility
// with Airbus CGS/CD-MCS system. It was redundant, because when false,
// the acquisitionStatus is also set to INVALID.
processingStatus: boolean;
monitoringResult: MonitoringResult;
rangeCondition: RangeCondition;

// Context-dependent ranges
alarmRange: AlarmRange[];

// How long (in milliseconds) this parameter value is valid
// Note that there is an option when subscribing to parameters to get
// updated when the parameter values expire.
expireMillis: string; // String decimal

// When transferring parameters over WebSocket, this value might be used
// instead of the id above in order to reduce the bandwidth.
// Note that the id <-> numericId assignment is only valid in the context
// of a single WebSocket call.
numericId: number;
}

// Used by external clients to identify an item in the Mission Database
// If namespace is set, then the name is that of an alias, rather than
// the qualified name.
interface NamedObjectId {
    name: string;
    namespace: string;
}

// Union type for storing a value
interface Value {
    type: Type;
    floatValue: number;
    doubleValue: number;
    sint32Value: number;
    uint32Value: number;
    binaryValue: string; // Base64
    stringValue: string;
    timestampValue: string; // String decimal
    uint64Value: string; // String decimal
    sint64Value: string; // String decimal
    booleanValue: boolean;
    aggregateValue: AggregateValue;
    arrayValue: Value[];
}

// An aggregate value is an ordered list of (member name, member value).
// Two arrays are used in order to be able to send just the values (since
// the names will not change)
interface AggregateValue {
    name: string[];
    value: Value[];
}

interface AlarmRange {
    level: AlarmLevelType;
    minInclusive: number;
    maxInclusive: number;
    minExclusive: number;
    maxExclusive: number;
}

```

```

enum Type {
    FLOAT = "FLOAT",
    DOUBLE = "DOUBLE",
    UINT32 = "UINT32",
    SINT32 = "SINT32",
    BINARY = "BINARY",
    STRING = "STRING",
    TIMESTAMP = "TIMESTAMP",
    UINT64 = "UINT64",
    SINT64 = "SINT64",
    BOOLEAN = "BOOLEAN",
    AGGREGATE = "AGGREGATE",
    ARRAY = "ARRAY",

    // Enumerated values have both an integer (sint64Value) and a string representation
    ENUMERATED = "ENUMERATED",
    NONE = "NONE",
}

enum AcquisitionStatus {

    // OK!
    ACQUIRED = "ACQUIRED",

    // No value received so far
    NOT_RECEIVED = "NOT_RECEIVED",

    // Some value has been received but is invalid
    INVALID = "INVALID",

    // The parameter is coming from a packet which has not since updated although it should have been
    EXPIRED = "EXPIRED",
}

enum MonitoringResult {
    DISABLED = "DISABLED",
    IN_LIMITS = "IN_LIMITS",
    WATCH = "WATCH",
    WARNING = "WARNING",
    DISTRESS = "DISTRESS",
    CRITICAL = "CRITICAL",
    SEVERE = "SEVERE",
}

enum RangeCondition {
    LOW = "LOW",
    HIGH = "HIGH",
}

enum AlarmLevelType {
    NORMAL = "NORMAL",
    WATCH = "WATCH",
    WARNING = "WARNING",
    DISTRESS = "DISTRESS",
    CRITICAL = "CRITICAL",
    SEVERE = "SEVERE",
}

```

26.3 Stream Parameter Values

Streams back parameter values

Warning: This method uses server-streaming. Yamcs sends an unspecified amount of data using chunked transfer encoding.

URI Template

```
POST /api/stream-archive/{instance}:streamParameterValues
```

{instance}

Request Body

```
interface StreamParameterValuesRequest {
  start: string; // RFC 3339 timestamp
  stop: string; // RFC 3339 timestamp
  ids: NamedObjectId[];
}
```

Response Type

```
interface ParameterData {
  parameter: ParameterValue[];

  // The next three fields are used by the recorder as unique key to store
  // parameters in "rows" and also by components that provide parameters
  // from external sources. The time should roughly correspond to the parameter
  // time but can be rounded for better efficiency.
  group: string;
  generationTime: string; // String decimal
  seqNum: number;

  // Used when parameter data is delivered as result of subscriptions
  subscriptionId: number;
}
```

Related Types

```
// Used by external clients to identify an item in the Mission Database
// If namespace is set, then the name is that of an alias, rather than
// the qualified name.
interface NamedObjectId {
  name: string;
  namespace: string;
}

interface ParameterValue {

  // Parameter identifier
  id: NamedObjectId;

  // Raw value (uncalibrated)
  rawValue: Value;

  // Engineering value (calibrated)
  engValue: Value;

  // Time of Yamcs reception
  acquisitionTime: string; // RFC 3339 timestamp

  // Time of generation (~ packet time)
  generationTime: string; // RFC 3339 timestamp
  acquisitionStatus: AcquisitionStatus;

  // Deprecated: this field was originally introduced for compatibility
  // with Airbus CGS/CD-MCS system. It was redundant, because when false,
```

(continues on next page)

```

// the acquisitionStatus is also set to INVALID.
processingStatus: boolean;
monitoringResult: MonitoringResult;
rangeCondition: RangeCondition;

// Context-dependent ranges
alarmRange: AlarmRange[];

// How long (in milliseconds) this parameter value is valid
// Note that there is an option when subscribing to parameters to get
// updated when the parameter values expire.
expireMillis: string; // String decimal

// When transferring parameters over WebSocket, this value might be used
// instead of the id above in order to reduce the bandwidth.
// Note that the id <-> numericId assignment is only valid in the context
// of a single WebSocket call.
numericId: number;
}

// Union type for storing a value
interface Value {
  type: Type;
  floatValue: number;
  doubleValue: number;
  sint32Value: number;
  uint32Value: number;
  binaryValue: string; // Base64
  stringValue: string;
  timestampValue: string; // String decimal
  uint64Value: string; // String decimal
  sint64Value: string; // String decimal
  booleanValue: boolean;
  aggregateValue: AggregateValue;
  arrayValue: Value[];
}

// An aggregate value is an ordered list of (member name, member value).
// Two arrays are used in order to be able to send just the values (since
// the names will not change)
interface AggregateValue {
  name: string[];
  value: Value[];
}

interface AlarmRange {
  level: AlarmLevelType;
  minInclusive: number;
  maxInclusive: number;
  minExclusive: number;
  maxExclusive: number;
}

enum Type {
  FLOAT = "FLOAT",
  DOUBLE = "DOUBLE",
  UINT32 = "UINT32",
  SINT32 = "SINT32",
  BINARY = "BINARY",
  STRING = "STRING",
  TIMESTAMP = "TIMESTAMP",
  UINT64 = "UINT64",
  SINT64 = "SINT64",
  BOOLEAN = "BOOLEAN",
  AGGREGATE = "AGGREGATE",
  ARRAY = "ARRAY",

  // Enumerated values have both an integer (sint64Value) and a string representation
  ENUMERATED = "ENUMERATED",
  NONE = "NONE",
}

```

```

enum AcquisitionStatus {

    // OK!
    ACQUIRED = "ACQUIRED",

    // No value received so far
    NOT_RECEIVED = "NOT_RECEIVED",

    // Some value has been received but is invalid
    INVALID = "INVALID",

    // The parameter is coming from a packet which has not since updated although it should have been
    EXPIRED = "EXPIRED",
}

enum MonitoringResult {
    DISABLED = "DISABLED",
    IN_LIMITS = "IN_LIMITS",
    WATCH = "WATCH",
    WARNING = "WARNING",
    DISTRESS = "DISTRESS",
    CRITICAL = "CRITICAL",
    SEVERE = "SEVERE",
}

enum RangeCondition {
    LOW = "LOW",
    HIGH = "HIGH",
}

enum AlarmLevelType {
    NORMAL = "NORMAL",
    WATCH = "WATCH",
    WARNING = "WARNING",
    DISTRESS = "DISTRESS",
    CRITICAL = "CRITICAL",
    SEVERE = "SEVERE",
}

```

26.4 Get Parameter Samples

Get parameter samples

URI Template

```
GET /api/stream-archive/{instance}/parameters/{name*}/samples
```

{instance}

Yamcs instance name.

{name*}

Parameter name.

Query Parameters

start

Filter the lower bound of the parameter's generation time. Specify a date string in ISO 8601 format.

stop

Filter the upper bound of the parameter's generation time. Specify a date string in ISO 8601 format.

count

Number of intervals to use. Default: 500.

norealtime

Disable loading of parameters from the parameter cache. Default: false.

useRawValue

Consider the raw value instead of the engineering value. Default is to use the engineering value

processor

The name of the processor from which to use the parameter cache. Default: realtime.

source

Specifies how to retrieve the parameters. Either ParameterArchive or replay. If replay is specified, a replay processor will be created and data will be processed with the active Mission Database. Note that this is much slower than receiving data from the ParameterArchive.

Default: ParameterArchive.

Response Type

```
interface TimeSeries {  
  sample: Sample[];  
}
```

Related Types

```
interface Sample {  
  time: string;  
  avg: number;  
  min: number;  
  max: number;  
  n: number;  
}
```

26.5 Export Parameter Values

Export parameter values in CSV format

Warning: This method uses server-streaming. Yamcs sends an unspecified amount of data using chunked transfer encoding.

URI Template

```
GET /api/archive/{instance}:exportParameterValues
```

{instance}

Yamcs instance name.

Query Parameters

start

Filter the lower bound of the parameter's generation time. Specify a date string in ISO 8601 format.

stop

Filter the upper bound of the parameter's generation time. Specify a date string in ISO 8601 format.

parameters

The parameters to add to the export.

namespace

Namespace used to display parameter names in csv header. Only used when no parameter ids were specified.

extra

Extra columns added to the CSV output:

- **raw**: Raw parameter values
- **monitoring**: Monitoring status

delimiter

Column delimiter. One of TAB, COMMA or SEMICOLON. Default: TAB.

27. Table

Service for reading and writing to Yamcs tables and streams

27.1 Execute Sql

Execute SQL

URI Template

```
POST /api/archive/{instance}:executeSql
```

{instance}

Yamcs instance name.

Request Body

```
interface ExecuteSqlRequest {  
  
    // StreamSQL statement  
    statement: string;  
}
```

Response Type

```
interface ResultSet {  
    columns: ColumnInfo[];  
    rows: ListValue[];  
}
```

Related Types

```
interface ColumnInfo {  
    name: string;  
    type: string;  
    enumValue: EnumValue[];  
}  
  
interface EnumValue {  
    value: number;  
    label: string;  
}  
  
interface ListValue {
```

(continues on next page)

```

    values: Value[];
}

// Union type for storing a value
interface Value {
    type: Type;
    floatValue: number;
    doubleValue: number;
    sint32Value: number;
    uint32Value: number;
    binaryValue: string; // Base64
    stringValue: string;
    timestampValue: string; // String decimal
    uint64Value: string; // String decimal
    sint64Value: string; // String decimal
    booleanValue: boolean;
    aggregateValue: AggregateValue;
    arrayValue: Value[];
}

// An aggregate value is an ordered list of (member name, member value).
// Two arrays are used in order to be able to send just the values (since
// the names will not change)
interface AggregateValue {
    name: string[];
    value: Value[];
}

enum Type {
    FLOAT = "FLOAT",
    DOUBLE = "DOUBLE",
    UINT32 = "UINT32",
    SINT32 = "SINT32",
    BINARY = "BINARY",
    STRING = "STRING",
    TIMESTAMP = "TIMESTAMP",
    UINT64 = "UINT64",
    SINT64 = "SINT64",
    BOOLEAN = "BOOLEAN",
    AGGREGATE = "AGGREGATE",
    ARRAY = "ARRAY",

    // Enumerated values have both an integer (sint64Value) and a string representation
    ENUMERATED = "ENUMERATED",
    NONE = "NONE",
}

```

27.2 Execute Streaming Sql

Execute streaming SQL

Warning: This method uses server-streaming. Yamcs sends an unspecified amount of data using chunked transfer encoding.

URI Template

POST /api/archive/{instance}:executeStreamingSql

{instance}

Yamcs instance name.

Request Body

```
interface ExecuteSqlRequest {  
  
    // StreamSQL statement  
    statement: string;  
}
```

Response Type

```
interface ResultSet {  
    columns: ColumnInfo[];  
    rows: ListValue[];  
}
```

Related Types

```
interface ColumnInfo {  
    name: string;  
    type: string;  
    enumValue: EnumValue[];  
}  
  
interface EnumValue {  
    value: number;  
    label: string;  
}  
  
interface ListValue {  
    values: Value[];  
}  
  
// Union type for storing a value  
interface Value {  
    type: Type;  
    floatValue: number;  
    doubleValue: number;  
    sint32Value: number;  
    uint32Value: number;  
    binaryValue: string; // Base64  
    stringValue: string;  
    timestampValue: string; // String decimal  
    uint64Value: string; // String decimal  
    sint64Value: string; // String decimal  
    booleanValue: boolean;  
    aggregateValue: AggregateValue;  
    arrayValue: Value[];  
}  
  
// An aggregate value is an ordered list of (member name, member value).  
// Two arrays are used in order to be able to send just the values (since  
// the names will not change)  
interface AggregateValue {  
    name: string[];  
    value: Value[];  
}  
  
enum Type {  
    FLOAT = "FLOAT",  
    DOUBLE = "DOUBLE",  
    UINT32 = "UINT32",  
    SINT32 = "SINT32",  
    BINARY = "BINARY",  
    STRING = "STRING",  
    TIMESTAMP = "TIMESTAMP",  
    UINT64 = "UINT64",
```

(continues on next page)

(continued from previous page)

```
SINT64 = "SINT64",
BOOLEAN = "BOOLEAN",
AGGREGATE = "AGGREGATE",
ARRAY = "ARRAY",

// Enumerated values have both an integer (sint64Value) and a string representation
ENUMERATED = "ENUMERATED",
NONE = "NONE",
}
```

27.3 List Streams

List streams

Note that this will only list the fixed columns of the stream. Tuples may always have extra columns.

URI Template

```
GET /api/archive/{instance}/streams
```

{instance}
Yamcs instance name.

Response Type

```
interface ListStreamsResponse {
    streams: StreamInfo[];
}
```

Related Types

```
interface StreamInfo {
    name: string;
    column: ColumnInfo[];
    script: string;
    dataCount: string; // String decimal
}

interface ColumnInfo {
    name: string;
    type: string;
    enumValue: EnumValue[];
}

interface EnumValue {
    value: number;
    label: string;
}
```

27.4 Subscribe Stream Statistics

Receive updates on stream stats

WebSocket

This method requires to upgrade an HTTP connection to WebSocket. See details on [how Yamcs uses WebSocket²²](#).

Use the message type stream-stats.

Input Type

```
interface SubscribeStreamStatisticsRequest {  
    instance: string;  
}
```

Output Type

```
interface StreamEvent {  
    type: Type;  
    name: string;  
    dataCount: string; // String decimal  
}
```

Related Types

```
enum Type {  
    CREATED = "CREATED",  
    DELETED = "DELETED",  
    UPDATED = "UPDATED",  
}
```

27.5 Get Stream

Get a stream

URI Template

```
GET /api/archive/{instance}/streams/{name}
```

{instance}

{name}

Response Type

```
interface StreamInfo {  
    name: string;  
    column: ColumnInfo[];  
    script: string;  
    dataCount: string; // String decimal  
}
```

²² <https://docs.yamcs.org/yamcs-http-api/websocket>

Related Types

```
interface ColumnInfo {
  name: string;
  type: string;
  enumValue: EnumValue[];
}

interface EnumValue {
  value: number;
  label: string;
}
```

27.6 Subscribe Stream

Receive stream updates

WebSocket

This method requires to upgrade an HTTP connection to WebSocket. See details on [how Yamcs uses WebSocket](#)²³.

Use the message type stream.

Input Type

```
interface SubscribeStreamRequest {
  instance: string;
  stream: string;
}
```

Output Type

```
interface StreamData {
  stream: string;
  column: ColumnData[];
}
```

Related Types

```
interface ColumnData {
  name: string;
  value: Value;
}

// Union type for storing a value
interface Value {
  type: Type;
  floatValue: number;
  doubleValue: number;
  sint32Value: number;
  uint32Value: number;
  binaryValue: string; // Base64
  stringValue: string;
  timestampValue: string; // String decimal
}
```

(continues on next page)

²³ <https://docs.yamcs.org/yamcs-http-api/websocket>

(continued from previous page)

```
uint64Value: string; // String decimal
sint64Value: string; // String decimal
booleanValue: boolean;
aggregateValue: AggregateValue;
arrayValue: Value[];
}

// An aggregate value is an ordered list of (member name, member value).
// Two arrays are used in order to be able to send just the values (since
// the names will not change)
interface AggregateValue {
    name: string[];
    value: Value[];
}

enum Type {
    FLOAT = "FLOAT",
    DOUBLE = "DOUBLE",
    UINT32 = "UINT32",
    SINT32 = "SINT32",
    BINARY = "BINARY",
    STRING = "STRING",
    TIMESTAMP = "TIMESTAMP",
    UINT64 = "UINT64",
    SINT64 = "SINT64",
    BOOLEAN = "BOOLEAN",
    AGGREGATE = "AGGREGATE",
    ARRAY = "ARRAY",

    // Enumerated values have both an integer (sint64Value) and a string representation
    ENUMERATED = "ENUMERATED",
    NONE = "NONE",
}
```

27.7 List Tables

List tables

The response will only include fixed columns of the table. Tuples may always add extra value columns.

URI Template

```
GET /api/archive/{instance}/tables
```

{instance}

Yamcs instance name.

Response Type

```
interface ListTablesResponse {
    tables: TableInfo[];
}
```

Related Types

```
interface TableInfo {
    name: string;
    keyColumn: ColumnInfo[];
    valueColumn: ColumnInfo[];
}
```

(continues on next page)

```

script: string;
histogramColumn: string[];
storageEngine: string;
formatVersion: number;
tablespace: string;
compressed: boolean;
partitioningInfo: PartitioningInfo;
}

interface ColumnInfo {
  name: string;
  type: string;
  enumValue: EnumValue[];
}

interface EnumValue {
  value: number;
  label: string;
}

interface PartitioningInfo {
  type: PartitioningType;
  timeColumn: string;
  timePartitionSchema: string;
  valueColumn: string;
  valueColumnType: string;
}

enum PartitioningType {
  TIME = "TIME",
  VALUE = "VALUE",
  TIME_AND_VALUE = "TIME_AND_VALUE",
}

```

27.8 Get Table

Get a table

URI Template

```
GET /api/archive/{instance}/tables/{name}
```

{instance}

Yamcs instance name.

{name}

Table name.

Response Type

```

interface TableInfo {
  name: string;
  keyColumn: ColumnInfo[];
  valueColumn: ColumnInfo[];
  script: string;
  histogramColumn: string[];
  storageEngine: string;
  formatVersion: number;
  tablespace: string;
  compressed: boolean;
  partitioningInfo: PartitioningInfo;
}

```


Related Types

```
interface ColumnInfo {
  name: string;
  type: string;
  enumValue: EnumValue[];
}

interface EnumValue {
  value: number;
  label: string;
}

interface PartitioningInfo {
  type: PartitioningType;
  timeColumn: string;
  timePartitionSchema: string;
  valueColumn: string;
  valueColumnType: string;
}

enum PartitioningType {
  TIME = "TIME",
  VALUE = "VALUE",
  TIME_AND_VALUE = "TIME_AND_VALUE",
}
```

27.9 Get Table Data

Get table data

URI Template

```
GET /api/archive/{instance}/tables/{name}/data
```

{instance}
Yamcs instance name.

{name}
Table name.

Query Parameters

cols

The columns to be included in the result. If unspecified, all table and/or additional tuple columns will be included.

pos

The zero-based row number at which to start outputting results. Default: 0 Note that in the current rocksdb storage engine there is no way to jump to a row by its number. This is why such a request will do a table scan and can be slow for large values of pos.

limit

The maximum number of returned records per page. Choose this value too high and you risk hitting the maximum response size limit enforced by the server. Default: 100

order

The direction of the sort. Sorting is always done on the key of the table. Can be either asc or desc.
Default: desc

Response Type

```
interface TableData {  
    record: TableRecord[];  
}
```

Related Types

```
interface TableRecord {  
    column: ColumnData[];  
}  
  
interface ColumnData {  
    name: string;  
    value: Value;  
}  
  
// Union type for storing a value  
interface Value {  
    type: Type;  
    floatValue: number;  
    doubleValue: number;  
    sint32Value: number;  
    uint32Value: number;  
    binaryValue: string; // Base64  
    stringValue: string;  
    timestampValue: string; // String decimal  
    uint64Value: string; // String decimal  
    sint64Value: string; // String decimal  
    booleanValue: boolean;  
    aggregateValue: AggregateValue;  
    arrayValue: Value[];  
}  
  
// An aggregate value is an ordered list of (member name, member value).  
// Two arrays are used in order to be able to send just the values (since  
// the names will not change)  
interface AggregateValue {  
    name: string[];  
    value: Value[];  
}  
  
enum Type {  
    FLOAT = "FLOAT",  
    DOUBLE = "DOUBLE",  
    UINT32 = "UINT32",  
    SINT32 = "SINT32",  
    BINARY = "BINARY",  
    STRING = "STRING",  
    TIMESTAMP = "TIMESTAMP",  
    UINT64 = "UINT64",  
    SINT64 = "SINT64",  
    BOOLEAN = "BOOLEAN",  
    AGGREGATE = "AGGREGATE",  
    ARRAY = "ARRAY",  
  
    // Enumerated values have both an integer (sint64Value) and a string representation  
    ENUMERATED = "ENUMERATED",  
    NONE = "NONE",  
}
```

27.10 Read Rows

Streams back the contents of all rows in key order

The ColumnInfo message assigns an integer id for each column and the id is present in each cell belonging to that column (this is done in order to avoid sending the ColumnInfo with each Cell). The column id starts from 0 and are incremented with each new column found. The ids are only valid during one single dump. The dumped data does not contain information on any table characteristics such as (primary) key, partitioning or other storage options.

Warning: This method uses server-streaming. Yamcs sends an unspecified amount of data using chunked transfer encoding.

URI Template

```
POST /api/archive/{instance}/tables/{table}:readRows
```

{instance}
Yamcs instance name.

{table}
Table name.

Request Body

```
interface ReadRowsRequest {  
  
    // The columns to be included in the result. If unspecified, all  
    // table and/or additional tuple columns will be included.  
    cols: string[];  
}
```

Response Type

```
interface Row {  
  
    //the column info is only present for new columns in a stream of Row messages  
    columns: ColumnInfo[];  
    cells: Cell[];  
}
```

Related Types

```
interface ColumnInfo {  
    id: number;  
    name: string;  
    type: string;  
  
    // The name of the class implementing the proto object if dataType is PROTOBUF  
    protoClass: string;  
}  
  
interface Cell {  
    columnId: number;  
    data: string; // Base64  
}
```

27.11 Write Rows

Imports a stream of rows

The table has to exist in order to load data into it.

As soon as the server detects an error with one of the written rows, it will forcefully close the connection and send back an early error message. The client should stop streaming and handle the error.

Note that the erratic condition causes the connection to be closed even if the Keep-Alive request header was enabled.

The error response is of type `ExceptionMessage` and contains a detail message of type `WriteRowsExceptionDetail` that provides the number of rows that were successfully written by the client. The client can use this information to link the error message to a row (i.e. the bad row is at position count + 1 of the stream).

One possible error could be that the table has defined a (primary) key and one of the loaded rows contains no value for one of the columns of the key.

The table load will overwrite any data existing in the table with the same key as the imported row.

The table load will not update the histograms so a histogram rebuild is required after the load.

Warning: This method uses client-streaming.

URI Template

```
POST /api/archive/{instance}/tables/{table}:writeRows
```

{instance}

Yamcs instance name.

{table}

Table name.

Request Body

```
interface Row {  
  
    //the column info is only present for new columns in a stream of Row messages  
    columns: ColumnInfo[];  
    cells: Cell[];  
}
```

Response Type

```
interface WriteRowsResponse {  
  
    // The number of rows that were written  
    count: number;  
}
```

Related Types

```

interface ColumnInfo {
    id: number;
    name: string;
    type: string;

    // The name of the class implementing the proto object if dataType is PROTOBUF
    protoClass: string;
}

interface Cell {
    columnId: number;
    data: string; // Base64
}

```

27.12 Rebuild Histogram

Rebuilds histograms - this is required after a table load.

Currently the time interval passed in the request will be used to select the partitions which will be rebuild - any partition overlapping with the interval will be rebuilt. If the table is not partitioned by time, the histogram for the entire table will be rebuild.

URI Template

```
POST /api/archive/{instance}/tables/{table}:rebuildHistogram
```

{instance}

Yamcs instance name.

{table}

Table name.

Request Body

```

interface RebuildHistogramRequest {

    // Specify a date string in ISO 8601 format. The histogram data for all partitions overlapping with the
    ↪ [start, stop] interval will be recreated.
    // If not specified, it is assumed as the start of the archive.
    start: string; // RFC 3339 timestamp

    // Specify a date string in ISO 8601 format. The histogram data for all partitions overlapping with the
    ↪ [start, stop] interval will be recreated.
    // If not specified, it is assumed as the end of the archive.
    stop: string; // RFC 3339 timestamp
}

```

Response Type

```

interface RebuildHistogramResponse {
}

```

28. Time Correlation

Methods related to the Time Correlation Service.

28.1 Get Config

Get the TCO config

Returns the TCO configuration comprising the accuracy, validity and the fixed delays.

URI Template

```
GET /api/tco/{instance}/{serviceName}/config
```

{instance}

Yamcs instance name.

{serviceName}

service name.

Response Type

```
interface TcoConfig {  
  accuracy: number;  
  validity: number;  
  onboardDelay: number;  
  defaultTof: number;  
}
```

28.2 Set Config

Set the TCO config

Set the TCO configuration. The configuration is not persisted on disk.

URI Template

```
POST /api/tco/{instance}/{serviceName}/config
```

{instance}

Yamcs instance name.

{serviceName}

Service name.

Request Body

```
interface TcoConfig {
    accuracy: number;
    validity: number;
    onboardDelay: number;
    defaultTof: number;
}
```

28.3 Get Status

Get the TCO status

Returns the TCO status comprising the currently used coefficients, the last computed deviation and the last received samples.

URI Template

```
GET /api/tco/{instance}/{serviceName}/status
```

{instance}
Yamcs instance name.

{serviceName}
Service name.

Response Type

```
//If the TCO is used only for verifying the synchronization, the message will
// contain only the validity, accuracy and deviation.
interface TcoStatus {

    //Currently used coefficients.
    // If not present, the synchronization is not established
    coefficients: TcoCoefficients;

    //The time when the coefficients have been computed
    coefficientsTime: string; // RFC 3339 timestamp

    //The last computed deviation
    deviation: number;

    //The last accumulated samples
    //These are not necessary those from which the coefficients
    //have been calculated because the coefficients will only
    //be recalculated when the deviation is higher than the accuracy settings
    samples: TcoSample[];
}
```

Related Types

```
interface TcoCoefficients {
    utc: string; // RFC 3339 timestamp
    obt: string; // String decimal
    gradient: number;
    offset: number;
}
```

//Sample association between UTC and Onboard time.

(continues on next page)

(continued from previous page)

```
//This is computed by the TCO service after adjusting for internal
//delays and time of flight.
interface TcoSample {
    utc: string; // RFC 3339 timestamp
    obt: string; // String decimal
}
```

28.4 Set Coefficients

Set the TCO coefficients

Manually set the coefficients to be used for time correlation. These will overwrite the automatic computed coefficients.

URI Template

```
POST /api/tco/{instance}/{serviceName}/coefficients
```

{instance}
Yamcs instance name.

{serviceName}
Service name.

Request Body

```
interface TcoCoefficients {
    utc: string; // RFC 3339 timestamp
    obt: string; // String decimal
    gradient: number;
    offset: number;
}
```

28.5 Reset

Reset the time correlation.

The current used TCO coefficients are removed together with all collected samples.

URI Template

```
POST /api/tco/{instance}/{serviceName}:reset
```

{instance}
Yamcs instance name.

{serviceName}
Service name.

28.6 Add Time Of Flight Intervals

Add intervals for the time of flight calculation.

Each [ertStart, ertStop) interval contains a polynomial function used to compute the time of flight for the given ert. The intervals can overlap and are sorted in descending order of the start time. The first (highest start time) interval where the requested ert fits, will be used for the calculation.

The formula used for calculating the time of flight for a frame/packet received at ert in the [ertStart, ertStop) interval is:

$$\text{delta} = \text{ert} - \text{ertStart}$$
$$\text{tof} = \text{polCoef}[0] + \text{polCoef}[1] * \text{delta} + \text{polCoef}[2] * \text{delta}^2 + \dots$$

The result of the polynomial is the tof expressed in seconds.

URI Template

```
POST /api/tco/{instance}/{serviceName}/tof:addIntervals
```

{instance}

Yamcs instance name.

{serviceName}

Service name.

Request Body

```
interface AddTimeOfFlightIntervalsRequest {  
    //intervals for time of flight calculation  
    intervals: TofInterval[];  
}
```

Related Types

```
interface TofInterval {  
    ertStart: string; // RFC 3339 timestamp  
    ertStop: string; // RFC 3339 timestamp  
    polCoef: number[];  
}
```

28.7 Delete Time Of Flight Intervals

Delete intervals for the time of flight calculation.

All the intervals with the start time falling in the requested [start, stop] interval will be removed.

URI Template

```
POST /api/tco/{instance}/{serviceName}/tof:deleteIntervals
```

{instance}

Yamcs instance name.

{serviceName}
Service name.

Request Body

```
//Delete all the TofIntervals having  
// start <= tofInterval.ertStart <= stop  
interface DeleteTimeOfFlightIntervalsRequest {  
    start: string; // RFC 3339 timestamp  
    stop: string; // RFC 3339 timestamp  
}
```

29. Time

29.1 Get Leap Seconds

Get UTC leap seconds

URI Template

```
GET /api/leap-seconds
```

Response Type

```
interface LeapSecondsTable {  
  ranges: ValidityRange[];  
}
```

Related Types

```
interface ValidityRange {  
  start: string;  
  stop: string;  
  leapSeconds: number;  
  taiDifference: number;  
}
```

29.2 Set Time

Set (simulation) time of an instance

URI Template

```
POST /api/instances/{instance}:setTime
```

{instance}

Request Body

```
interface SetTimeRequest {
    time0: string; // RFC 3339 timestamp
    elapsedTime: string; // String decimal
    speed: number;
}
```

29.3 Subscribe Time

Receive time updates

WebSocket

This method requires to upgrade an HTTP connection to WebSocket. See details on [how Yamcs uses WebSocket²⁴](#).

Use the message type time.

Input Type

```
interface SubscribeTimeRequest {
    instance: string;
    processor: string;
}
```

Output Type

```
// A Timestamp represents a point in time independent of any time zone or local
// calendar, encoded as a count of seconds and fractions of seconds at
// nanosecond resolution. The count is relative to an epoch at UTC midnight on
// January 1, 1970, in the proleptic Gregorian calendar which extends the
// Gregorian calendar backwards to year one.
//
// All minutes are 60 seconds long. Leap seconds are "smeared" so that no leap
// second table is needed for interpretation, using a [24-hour linear
// smear](https://developers.google.com/time/smear).
//
// The range is from 0001-01-01T00:00:00Z to 9999-12-31T23:59:59.999999999Z. By
// restricting to that range, we ensure that we can convert to and from [RFC
// 3339](https://www.ietf.org/rfc/rfc3339.txt) date strings.
//
// # Examples
//
// Example 1: Compute Timestamp from POSIX `time()`.
//
//     Timestamp timestamp;
//     timestamp.set_seconds(time(NULL));
//     timestamp.set_nanos(0);
//
// Example 2: Compute Timestamp from POSIX `gettimeofday()`.
//
//     struct timeval tv;
//     gettimeofday(&tv, NULL);
//
//     Timestamp timestamp;
//     timestamp.set_seconds(tv.tv_sec);
//     timestamp.set_nanos(tv.tv_usec * 1000);
//
// Example 3: Compute Timestamp from Win32 `GetSystemTimeAsFileTime()`.
//
```

(continues on next page)

²⁴ <https://docs.yamcs.org/yamcs-http-api/websocket>

```

//
// FILETIME ft;
// GetSystemTimeAsFileTime(&ft);
// UINT64 ticks = (((UINT64)ft.dwHighDateTime) << 32) | ft.dwLowDateTime;
//
// // A Windows tick is 100 nanoseconds. Windows epoch 1601-01-01T00:00:00Z
// // is 11644473600 seconds before Unix epoch 1970-01-01T00:00:00Z.
// Timestamp timestamp;
// timestamp.set_seconds((INT64) ((ticks / 100000000) - 11644473600LL));
// timestamp.set_nanos((INT32) ((ticks % 100000000) * 100));
//
// Example 4: Compute Timestamp from Java `System.currentTimeMillis()`.
//
// long millis = System.currentTimeMillis();
//
// Timestamp timestamp = Timestamp.newBuilder().setSeconds(millis / 1000)
// .setNanos((int) ((millis % 1000) * 100000000)).build();
//
//
// Example 5: Compute Timestamp from current time in Python.
//
// timestamp = Timestamp()
// timestamp.GetCurrentTime()
//
// # JSON Mapping
//
// In JSON format, the Timestamp type is encoded as a string in the
// [RFC 3339](https://www.ietf.org/rfc/rfc3339.txt) format. That is, the
// format is "{year}-{month}-{day}T{hour}:{min}:{sec}[.{frac_sec}]Z"
// where {year} is always expressed using four digits while {month}, {day},
// {hour}, {min}, and {sec} are zero-padded to two digits each. The fractional
// seconds, which can go up to 9 digits (i.e. up to 1 nanosecond resolution),
// are optional. The "Z" suffix indicates the timezone ("UTC"); the timezone
// is required. A proto3 JSON serializer should always use UTC (as indicated by
// "Z") when printing the Timestamp type and a proto3 JSON parser should be
// able to accept both UTC and other timezones (as indicated by an offset).
//
// For example, "2017-01-15T01:30:15.01Z" encodes 15.01 seconds past
// 01:30 UTC on January 15, 2017.
//
// In JavaScript, one can convert a Date object to this format using the
// standard
// [toISOString()](https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/Date/
// toISOString)
// method. In Python, a standard `datetime.datetime` object can be converted
// to this format using
// [strftime](https://docs.python.org/2/library/time.html#time.strftime) with
// the time format spec '%Y-%m-%dT%H:%M:%S.%fZ'. Likewise, in Java, one can use
// the Joda Time's [ISODateTimeFormat.dateTime()](
// http://www.joda.org/joda-time/apidocs/org/joda/time/format/ISODateTimeFormat.html#dateTime%2D%2D
// ) to obtain a formatter capable of generating timestamps in this format.
interface Timestamp {

    // Represents seconds of UTC time since Unix epoch
    // 1970-01-01T00:00:00Z. Must be from 0001-01-01T00:00:00Z to
    // 9999-12-31T23:59:59Z inclusive.
    seconds: string; // String decimal

    // Non-negative fractions of a second at nanosecond resolution. Negative
    // second values with fractions must still have non-negative nanos values
    // that count forward in time. Must be from 0 to 999,999,999
    // inclusive.
    nanos: number;
}

```

30. Timeline

Methods related to the Timeline Service.

30.1 Create Item

Create an item

URI Template

```
POST /api/timeline/{instance}/items
```

{instance}

Yamcs instance name

Request Body

```
//***** Items *****  
interface CreateItemRequest {  
  
    // Item source  
    source: string;  
  
    // Item name  
    name: string;  
  
    // Type of item  
    type: TimelineItemType;  
  
    //item start. The start and the relativeTime (below) are mutually exclusive  
    start: string; // RFC 3339 timestamp  
  
    //item duration. Applies also if the relativeTime is specified  
    duration: string; // Duration in seconds. Example: "3s" or "3.001s"  
  
    //tags  
    tags: string[];  
  
    //if this item is part of a group, this is the group identifier  
    groupId: string;  
  
    //if this item time specification is relative to another item, relativeTime contains a reference  
    // to that item as well as the relative start (the duration is the same as given by the duration above)  
    relativeTime: RelativeTime;  
}
```

Response Type

```
interface TimelineItem {

    // Item identifier
    id: string;

    // Item name
    name: string;
    type: TimelineItemType;
    start: string; // RFC 3339 timestamp
    duration: string; // Duration in seconds. Example: "3s" or "3.001s"
    tags: string[];

    //if this item is part of a group, this is the group identifier
    groupId: string;

    //if this item time specification is relative to another item, relativeTime contains a reference
    // to that item as well as the relative start (the duration is the same as given by the duration above)
    //note that start of the item will be computed by the server based on the relativeTime before sending ↵
    ↵ the item to the client
    relativeTime: RelativeTime;
}
```

Related Types

```
interface RelativeTime {

    // Identifier of the item that this time is relative to.
    relto: string;
    relativeStart: string; // Duration in seconds. Example: "3s" or "3.001s"
}

enum TimelineItemType {

    //events are the most generic timeline items.
    EVENT = "EVENT",

    //unlike events, activities have an execution status
    //manual activity's execution status is controlled by an operator
    MANUAL_ACTIVITY = "MANUAL_ACTIVITY",

    //activity which is automatically executed on the server (the status changes automatically)
    AUTO_ACTIVITY = "AUTO_ACTIVITY",

    //a grouping of other items (events and/or activities)
    ITEM_GROUP = "ITEM_GROUP",

    //a grouping of activities - unlike the ITEM_GROUP, this group in itself is an automated activity
    ACTIVITY_GROUP = "ACTIVITY_GROUP",
}
```

30.2 Get Item

Get an item

URI Template

```
GET /api/timeline/{instance}/items/{id}
```

{instance}

Yamcs instance name

{id}
Item identifier

Query Parameters

source

Item source

Response Type

```
interface TimelineItem {  
  
    // Item identifier  
    id: string;  
  
    // Item name  
    name: string;  
    type: TimelineItemType;  
    start: string; // RFC 3339 timestamp  
    duration: string; // Duration in seconds. Example: "3s" or "3.001s"  
    tags: string[];  
  
    //if this item is part of a group, this is the group identifier  
    groupId: string;  
  
    //if this item time specification is relative to another item, relativeTime contains a reference  
    // to that item as well as the relative start (the duration is the same as given by the duration above)  
    //note that start of the item will be computed by the server based on the relativeTime before sending.  
    //the item to the client  
    relativeTime: RelativeTime;  
}
```

Related Types

```
interface RelativeTime {  
  
    // Identifier of the item that this time is relative to.  
    relto: string;  
    relativeStart: string; // Duration in seconds. Example: "3s" or "3.001s"  
}
```

```
enum TimelineItemType {  
  
    //events are the most generic timeline items.  
    EVENT = "EVENT",  
  
    //unlike events, activities have an execution status  
    //manual activity's execution status is controlled by an operator  
    MANUAL_ACTIVITY = "MANUAL_ACTIVITY",  
  
    //activity which is automatically executed on the server (the status changes automatically)  
    AUTO_ACTIVITY = "AUTO_ACTIVITY",  
  
    //a grouping of other items (events and/or activities)  
    ITEM_GROUP = "ITEM_GROUP",  
  
    //a grouping of activities - unlike the ITEM_GROUP, this group in itself is an automated activity  
    ACTIVITY_GROUP = "ACTIVITY_GROUP",  
}
```


30.3 Update Item

Update an item

URI Template

```
PUT /api/timeline/{instance}/items/{id}
```

{instance}

Yamcs instance name

{id}

Item identifier

Request Body

```
interface UpdateItemRequest {  
  
    // Item source  
    source: string;  
  
    // Item name  
    name: string;  
  
    //new start time  
    start: string; // RFC 3339 timestamp  
  
    //new duration  
    duration: string; // Duration in seconds. Example: "3s" or "3.001s"  
  
    //new tags  
    tags: string[];  
  
    //new group identifier  
    groupId: string;  
  
    //new relative time  
    relativeTime: RelativeTime;  
}
```

Response Type

```
interface TimelineItem {  
  
    // Item identifier  
    id: string;  
  
    // Item name  
    name: string;  
    type: TimelineItemType;  
    start: string; // RFC 3339 timestamp  
    duration: string; // Duration in seconds. Example: "3s" or "3.001s"  
    tags: string[];  
  
    //if this item is part of a group, this is the group identifier  
    groupId: string;  
  
    //if this item time specification is relative to another item, relativeTime contains a reference  
    // to that item as well as the relative start (the duration is the same as given by the duration above)  
    //note that start of the item will be computed by the server based on the relativeTime before sending  
    //the item to the client  
    relativeTime: RelativeTime;  
}
```

Related Types

```
interface RelativeTime {  
  
    // Identifier of the item that this time is relative to.  
    relto: string;  
    relativeStart: string; // Duration in seconds. Example: "3s" or "3.001s"  
}  
  
enum TimelineItemType {  
  
    //events are the most generic timeline items.  
    EVENT = "EVENT",  
  
    //unlike events, activities have an execution status  
    //manual activity's execution status is controlled by an operator  
    MANUAL_ACTIVITY = "MANUAL_ACTIVITY",  
  
    //activity which is automatically executed on the server (the status changes automatically)  
    AUTO_ACTIVITY = "AUTO_ACTIVITY",  
  
    //a grouping of other items (events and/or activities)  
    ITEM_GROUP = "ITEM_GROUP",  
  
    //a grouping of activities - unlike the ITEM_GROUP, this group in itself is an automated activity  
    ACTIVITY_GROUP = "ACTIVITY_GROUP",  
}
```

30.4 List Items

List items

URI Template

```
GET /api/timeline/{instance}/items
```

{instance}
Yamcs instance name

Query Parameters

source

Item source

limit

next

Continuation token returned by a previous page response.

start

stop

band

Include only items matching a specific band's tags

Response Type

```
interface ListItemsResponse {
    items: TimelineItem[];

    // Token indicating the response is only partial. More results can then
    // be obtained by performing the same request (including all original
    // query parameters) and setting the ``next`` parameter to this token.
    continuationToken: string;
}
```

Related Types

```
interface TimelineItem {

    // Item identifier
    id: string;

    // Item name
    name: string;
    type: TimelineItemType;
    start: string; // RFC 3339 timestamp
    duration: string; // Duration in seconds. Example: "3s" or "3.001s"
    tags: string[];

    //if this item is part of a group, this is the group identifier
    groupId: string;

    //if this item time specification is relative to another item, relativeTime contains a reference
    // to that item as well as the relative start (the duration is the same as given by the duration above)
    //note that start of the item will be computed by the server based on the relativeTime before sending.
    //the item to the client
    relativeTime: RelativeTime;
}

interface RelativeTime {

    // Identifier of the item that this time is relative to.
    relto: string;
    relativeStart: string; // Duration in seconds. Example: "3s" or "3.001s"
}

enum TimelineItemType {

    //events are the most generic timeline items.
    EVENT = "EVENT",

    //unlike events, activities have an execution status
    //manual activity's execution status is controlled by an operator
    MANUAL_ACTIVITY = "MANUAL_ACTIVITY",

    //activity which is automatically executed on the server (the status changes automatically)
    AUTO_ACTIVITY = "AUTO_ACTIVITY",

    //a grouping of other items (events and/or activities)
    ITEM_GROUP = "ITEM_GROUP",

    //a grouping of activities - unlike the ITEM_GROUP, this group in itself is an automated activity
    ACTIVITY_GROUP = "ACTIVITY_GROUP",
}
```

30.5 Delete Item

Delete an item

URI Template

```
DELETE /api/timeline/{instance}/items/{id}
```

{instance}

Yamcs instance name.

{id}

Item identifier

Response Type

```
interface TimelineItem {  
  
    // Item identifier  
    id: string;  
  
    // Item name  
    name: string;  
    type: TimelineItemType;  
    start: string; // RFC 3339 timestamp  
    duration: string; // Duration in seconds. Example: "3s" or "3.001s"  
    tags: string[];  
  
    //if this item is part of a group, this is the group identifier  
    groupId: string;  
  
    //if this item time specification is relative to another item, relativeTime contains a reference  
    // to that item as well as the relative start (the duration is the same as given by the duration above)  
    //note that start of the item will be computed by the server based on the relativeTime before sending  
    //the item to the client  
    relativeTime: RelativeTime;  
}
```

Related Types

```
interface RelativeTime {  
  
    // Identifier of the item that this time is relative to.  
    relto: string;  
    relativeStart: string; // Duration in seconds. Example: "3s" or "3.001s"  
}  
  
enum TimelineItemType {  
  
    //events are the most generic timeline items.  
    EVENT = "EVENT",  
  
    //unlike events, activities have an execution status  
    //manual activity's execution status is controlled by an operator  
    MANUAL_ACTIVITY = "MANUAL_ACTIVITY",  
  
    //activity which is automatically executed on the server (the status changes automatically)  
    AUTO_ACTIVITY = "AUTO_ACTIVITY",  
  
    //a grouping of other items (events and/or activities)  
    ITEM_GROUP = "ITEM_GROUP",  
  
    //a grouping of activities - unlike the ITEM_GROUP, this group in itself is an automated activity
```

(continues on next page)

```
ACTIVITY_GROUP = "ACTIVITY_GROUP",
}
```

30.6 Delete Timeline Group

Delete a group

URI Template

```
DELETE /api/timeline/{instance}/groups/{id}
```

{instance}

Yamcs instance name.

{id}

Group identifier

Response Type

```
interface TimelineItem {

    // Item identifier
    id: string;

    // Item name
    name: string;
    type: TimelineItemType;
    start: string; // RFC 3339 timestamp
    duration: string; // Duration in seconds. Example: "3s" or "3.001s"
    tags: string[];

    //if this item is part of a group, this is the group identifier
    groupId: string;

    //if this item time specification is relative to another item, relativeTime contains a reference
    // to that item as well as the relative start (the duration is the same as given by the duration above)
    //note that start of the item will be computed by the server based on the relativeTime before sending
    //the item to the client
    relativeTime: RelativeTime;
}
```

Related Types

```
interface RelativeTime {

    // Identifier of the item that this time is relative to.
    relto: string;
    relativeStart: string; // Duration in seconds. Example: "3s" or "3.001s"
}

enum TimelineItemType {

    //events are the most generic timeline items.
    EVENT = "EVENT",

    //unlike events, activities have an execution status
    //manual activity's execution status is controlled by an operator
    MANUAL_ACTIVITY = "MANUAL_ACTIVITY",
```

(continues on next page)

(continued from previous page)

```
//activity which is automatically executed on the server (the status changes automatically)
AUTO_ACTIVITY = "AUTO_ACTIVITY",

//a grouping of other items (events and/or activities)
ITEM_GROUP = "ITEM_GROUP",

//a grouping of activities - unlike the ITEM_GROUP, this group in itself is an automated activity
ACTIVITY_GROUP = "ACTIVITY_GROUP",
}
```

30.7 List Sources

List timeline sources

Usually there is a source named 'rdb' which provides items from an internal RocksDB database. Other external sources may be created by adding plugins (e.g. a shift planner)

URI Template

```
GET /api/timeline/{instance}/sources
```

{instance}

Yamcs instance name

Response Type

```
interface ListSourcesResponse {
  sources: {[key: string]: TimelineSourceCapabilities};
}
```

30.8 List Tags

List all tags available for a source

Note that currently the 'rdb' source does not discard unused tags (e.g. if all item using one tag have been deleted, the tag will still be returned by this call)

URI Template

```
GET /api/timeline/{instance}/tags
```

{instance}

Yamcs instance name

Query Parameters

source

Response Type

```
interface ListTimelineTagsResponse {  
    tags: string[];  
}
```

30.9 Add Band

Add a band. The band must not have the id set.

URI Template

```
POST /api/timeline/{instance}/bands
```

{instance}
Yamcs instance name

Request Body

```
/** ***** Bands ***** */  
interface AddBandRequest {  
  
    // Band name  
    name: string;  
  
    //if true, all users have access to this band, otherwise only the user who has created it  
    shared: boolean;  
  
    //the band contains only items from this source  
    source: string;  
  
    // Items containing these tags will be part of the timeline  
    tags: string[];  
  
    // Type of band  
    type: TimelineBandType;  
  
    // Band description  
    description: string;  
  
    // Additional properties used by yamcs-web to render this band  
    properties: {[key: string]: string};  
}
```

Response Type

```
interface TimelineBand {  
  
    // Yamcs instance name  
    instance: string;  
  
    // Band identifier  
    id: string;  
  
    // Band name  
    name: string;  
  
    //user who has created the band  
    username: string;
```

(continues on next page)

(continued from previous page)

```
//if true, all users have access to this band, otherwise only the user who has created it
shared: boolean;

//the band contains only items from this source
source: string;

//the band contains only items with these tags; if the list is empty, then all items from the given
↪source are part of the band
tags: string[];

// Type of band
type: TimelineBandType;

// Band description
description: string;

// Additional properties used by yamcs-web to render this band
properties: {[key: string]: string};
}
```

Related Types

```
enum TimelineBandType {
    TIME_RULER = "TIME_RULER",
    ITEM_BAND = "ITEM_BAND",
    SPACER = "SPACER",
    COMMAND_BAND = "COMMAND_BAND",
}
```

30.10 Get Band

Get a band

URI Template

```
GET /api/timeline/{instance}/bands/{id}
```

{instance}

Yamcs instance name

{id}

Item identifier

Response Type

```
interface TimelineBand {

    // Yamcs instance name
    instance: string;

    // Band identifier
    id: string;

    // Band name
    name: string;

    //user who has created the band
```

(continues on next page)

(continued from previous page)

```
username: string;

//if true, all users have access to this band, otherwise only the user who has created it
shared: boolean;

//the band contains only items from this source
source: string;

//the band contains only items with these tags; if the list is empty, then all items from the given_
↪source are part of the band
tags: string[];

// Type of band
type: TimelineBandType;

// Band description
description: string;

// Additional properties used by yamcs-web to render this band
properties: {[key: string]: string};
}
```

Related Types

```
enum TimelineBandType {
    TIME_RULER = "TIME_RULER",
    ITEM_BAND = "ITEM_BAND",
    SPACER = "SPACER",
    COMMAND_BAND = "COMMAND_BAND",
}
```

30.11 List Bands

List all bands

URI Template

```
GET /api/timeline/{instance}/bands
```

{instance}

Yamcs instance name

Response Type

```
interface ListBandsResponse {
    bands: TimelineBand[];
}
```

Related Types

```
interface TimelineBand {

    // Yamcs instance name
    instance: string;

    // Band identifier
```

(continues on next page)

(continued from previous page)

```
id: string;

// Band name
name: string;

//user who has created the band
username: string;

//if true, all users have access to this band, otherwise only the user who has created it
shared: boolean;

//the band contains only items from this source
source: string;

//the band contains only items with these tags; if the list is empty, then all items from the given
↪source are part of the band
tags: string[];

// Type of band
type: TimelineBandType;

// Band description
description: string;

// Additional properties used by yamcs-web to render this band
properties: {[key: string]: string};
}

enum TimelineBandType {
    TIME_RULER = "TIME_RULER",
    ITEM_BAND = "ITEM_BAND",
    SPACER = "SPACER",
    COMMAND_BAND = "COMMAND_BAND",
}
```

30.12 Update Band

Update a band

URI Template

```
PUT /api/timeline/{instance}/bands/{id}
```

{instance}

Yamcs instance name

{id}

Band identifier

Request Body

```
interface UpdateBandRequest {

    // Band name
    name: string;

    // Band description
    description: string;

    //if true, all users have access to this band, otherwise only the user who has created it
    shared: boolean;
```

(continues on next page)

```
// Items containing these tags will be part of the timeline
tags: string[];

// Additional properties used by yamcs-web to render this band
properties: {[key: string]: string};
}
```

Response Type

```
interface TimelineBand {

    // Yamcs instance name
    instance: string;

    // Band identifier
    id: string;

    // Band name
    name: string;

    //user who has created the band
    username: string;

    //if true, all users have access to this band, otherwise only the user who has created it
    shared: boolean;

    //the band contains only items from this source
    source: string;

    //the band contains only items with these tags; if the list is empty, then all items from the given
    ↪source are part of the band
    tags: string[];

    // Type of band
    type: TimelineBandType;

    // Band description
    description: string;

    // Additional properties used by yamcs-web to render this band
    properties: {[key: string]: string};
}
```

Related Types

```
enum TimelineBandType {
    TIME_RULER = "TIME_RULER",
    ITEM_BAND = "ITEM_BAND",
    SPACER = "SPACER",
    COMMAND_BAND = "COMMAND_BAND",
}
```

30.13 Delete Band

Delete a band

URI Template

```
DELETE /api/timeline/{instance}/bands/{id}
```

{instance}

Yamcs instance name.

{id}

Item identifier

Response Type

```
interface TimelineBand {  
  
    // Yamcs instance name  
    instance: string;  
  
    // Band identifier  
    id: string;  
  
    // Band name  
    name: string;  
  
    //user who has created the band  
    username: string;  
  
    //if true, all users have access to this band, otherwise only the user who has created it  
    shared: boolean;  
  
    //the band contains only items from this source  
    source: string;  
  
    //the band contains only items with these tags; if the list is empty, then all items from the given  
    ↪source are part of the band  
    tags: string[];  
  
    // Type of band  
    type: TimelineBandType;  
  
    // Band description  
    description: string;  
  
    // Additional properties used by yamcs-web to render this band  
    properties: {[key: string]: string};  
}
```

Related Types

```
enum TimelineBandType {  
    TIME_RULER = "TIME_RULER",  
    ITEM_BAND = "ITEM_BAND",  
    SPACER = "SPACER",  
    COMMAND_BAND = "COMMAND_BAND",  
}
```

30.14 Add View

Add a view

URI Template

```
POST /api/timeline/{instance}/views
```

{instance}

Yamcs instance name

Request Body

```
/** ***** Views ***** */
interface AddViewRequest {

    // View name
    name: string;

    // View description
    description: string;

    // The bands from this view
    bands: string[];
}
```

Response Type

```
interface TimelineView {

    // Yamcs instance name
    instance: string;

    // View identifier
    id: string;

    // View name
    name: string;

    // View description
    description: string;

    // array of bands
    bands: TimelineBand[];
}
```

Related Types

```
interface TimelineBand {

    // Yamcs instance name
    instance: string;

    // Band identifier
    id: string;

    // Band name
    name: string;

    //user who has created the band
    username: string;

    //if true, all users have access to this band, otherwise only the user who has created it
    shared: boolean;

    //the band contains only items from this source
}
```

(continues on next page)

(continued from previous page)

```
source: string;

//the band contains only items with these tags; if the list is empty, then all items from the given
↪source are part of the band
tags: string[];

// Type of band
type: TimelineBandType;

// Band description
description: string;

// Additional properties used by yamcs-web to render this band
properties: {[key: string]: string};
}

enum TimelineBandType {
    TIME_RULER = "TIME_RULER",
    ITEM_BAND = "ITEM_BAND",
    SPACER = "SPACER",
    COMMAND_BAND = "COMMAND_BAND",
}
```

30.15 Get View

Get a view

URI Template

```
GET /api/timeline/{instance}/views/{id}
```

{instance}
Yamcs instance name

{id}
Item identifier

Response Type

```
interface TimelineView {

    // Yamcs instance name
    instance: string;

    // View identifier
    id: string;

    // View name
    name: string;

    // View description
    description: string;

    // array of bands
    bands: TimelineBand[];
}
```

Related Types

```
interface TimelineBand {  
  
    // Yamcs instance name  
    instance: string;  
  
    // Band identifier  
    id: string;  
  
    // Band name  
    name: string;  
  
    //user who has created the band  
    username: string;  
  
    //if true, all users have access to this band, otherwise only the user who has created it  
    shared: boolean;  
  
    //the band contains only items from this source  
    source: string;  
  
    //the band contains only items with these tags; if the list is empty, then all items from the given  
    ↪source are part of the band  
    tags: string[];  
  
    // Type of band  
    type: TimelineBandType;  
  
    // Band description  
    description: string;  
  
    // Additional properties used by yamcs-web to render this band  
    properties: {[key: string]: string};  
}  
  
enum TimelineBandType {  
    TIME_RULER = "TIME_RULER",  
    ITEM_BAND = "ITEM_BAND",  
    SPACER = "SPACER",  
    COMMAND_BAND = "COMMAND_BAND",  
}
```

30.16 List Views

List all views

URI Template

```
GET /api/timeline/{instance}/views
```

{instance}
Yamcs instance name

Response Type

```
interface ListViewsResponse {  
    views: TimelineView[];  
}
```

Related Types

```
interface TimelineView {  
  
    // Yamcs instance name  
    instance: string;  
  
    // View identifier  
    id: string;  
  
    // View name  
    name: string;  
  
    // View description  
    description: string;  
  
    // array of bands  
    bands: TimelineBand[];  
}  
  
interface TimelineBand {  
  
    // Yamcs instance name  
    instance: string;  
  
    // Band identifier  
    id: string;  
  
    // Band name  
    name: string;  
  
    //user who has created the band  
    username: string;  
  
    //if true, all users have access to this band, otherwise only the user who has created it  
    shared: boolean;  
  
    //the band contains only items from this source  
    source: string;  
  
    //the band contains only items with these tags; if the list is empty, then all items from the given  
    ↪source are part of the band  
    tags: string[];  
  
    // Type of band  
    type: TimelineBandType;  
  
    // Band description  
    description: string;  
  
    // Additional properties used by yamcs-web to render this band  
    properties: {[key: string]: string};  
}  
  
enum TimelineBandType {  
    TIME_RULER = "TIME_RULER",  
    ITEM_BAND = "ITEM_BAND",  
    SPACER = "SPACER",  
    COMMAND_BAND = "COMMAND_BAND",  
}
```

30.17 Update View

Update a view

URI Template

```
PUT /api/timeline/{instance}/views/{id}
```

{instance}

Yamcs instance name

{id}

View identifier

Request Body

```
interface UpdateViewRequest {  
  
    // View name  
    name: string;  
  
    // View description  
    description: string;  
  
    // The bands from this view  
    bands: string[];  
}
```

Response Type

```
interface TimelineView {  
  
    // Yamcs instance name  
    instance: string;  
  
    // View identifier  
    id: string;  
  
    // View name  
    name: string;  
  
    // View description  
    description: string;  
  
    // array of bands  
    bands: TimelineBand[];  
}
```

Related Types

```
interface TimelineBand {  
  
    // Yamcs instance name  
    instance: string;  
  
    // Band identifier  
    id: string;  
  
    // Band name  
    name: string;  
  
    //user who has created the band  
    username: string;  
  
    //if true, all users have access to this band, otherwise only the user who has created it  
    shared: boolean;
```

(continues on next page)

```

//the band contains only items from this source
source: string;

//the band contains only items with these tags; if the list is empty, then all items from the given
↪source are part of the band
tags: string[];

// Type of band
type: TimelineBandType;

// Band description
description: string;

// Additional properties used by yamcs-web to render this band
properties: {[key: string]: string};
}

enum TimelineBandType {
    TIME_RULER = "TIME_RULER",
    ITEM_BAND = "ITEM_BAND",
    SPACER = "SPACER",
    COMMAND_BAND = "COMMAND_BAND",
}

```

30.18 Delete View

Delete a view

URI Template

```
DELETE /api/timeline/{instance}/views/{id}
```

{instance}

Yamcs instance name.

{id}

Item identifier

Response Type

```

interface TimelineView {

    // Yamcs instance name
    instance: string;

    // View identifier
    id: string;

    // View name
    name: string;

    // View description
    description: string;

    // array of bands
    bands: TimelineBand[];
}

```

Related Types

```
interface TimelineBand {  
  
    // Yamcs instance name  
    instance: string;  
  
    // Band identifier  
    id: string;  
  
    // Band name  
    name: string;  
  
    //user who has created the band  
    username: string;  
  
    //if true, all users have access to this band, otherwise only the user who has created it  
    shared: boolean;  
  
    //the band contains only items from this source  
    source: string;  
  
    //the band contains only items with these tags; if the list is empty, then all items from the given  
    ↪source are part of the band  
    tags: string[];  
  
    // Type of band  
    type: TimelineBandType;  
  
    // Band description  
    description: string;  
  
    // Additional properties used by yamcs-web to render this band  
    properties: {[key: string]: string};  
}  
  
enum TimelineBandType {  
    TIME_RULER = "TIME_RULER",  
    ITEM_BAND = "ITEM_BAND",  
    SPACER = "SPACER",  
    COMMAND_BAND = "COMMAND_BAND",  
}
```