Q:marketing Aktiengesellschaft
Düsseldorfer Straße 193
45481 Mülheim an der Ruhr
Telefon (02 08) 30 15 0
Telefax (02 08) 30 15 555
Internet: www.Qmarketing.de
dialog@Qmarketing.de

DESTO/DEBJE

Events

1 Requirements

The connecting portal client has requested a signed certificate from the portal server and the signed certificate is available locally

2 API endpoints

The API for pushing events to the portal server will be available via HTTPS and websocket over HTTPS.

The portal client must supply its certificate to the portal server when establishing a connection. If the portal client doesn't send its certificate to the portal server or the certificate is invalid (revoked, not signed by the ISPf Root CA), the portal server will respond with the HTTP status code 403 "Forbidden".

3 Event format

3.1.1 Data format

The data format for events will be JSON.

3.1.2 Required information in event format

3.1.2.1 ID

The portal client must generate an id for every event. The id must be an UUID (http://en.wikipedia.org/wiki/Universally_unique_identifier) (Version 4).

3.1.2.2 Timestamp

The portal client must supply a local timestamp for the event. The timestamp must be compatible to the ISO 8601 format (http://en.wikipedia.org/wiki/ISO_8601).

3.1.2.3 Type

The portal client must supply the type of the event. See document „Event types“ for a list of available events.

3.1.3 Optional information in event format

3.1.3.1 Belongs-To

The portal client can supply an optional reference to another event within „belongs-to“.

3.1.3.2 Payload

The portal client can supply an optional „payload“ for the event (textual information, JSON, XML, XML-RPC, images, ...). Given, that the chosen data format, JSON, doesn't support multiline strings for values, the payload must be encoded with Base64.

3.1.3.3 Destination

The portal client can supply an array of recipient-UUIDs for the event. See chapter „Discovery“ of document „Event types“.

If no value for an optional field is present, the optional field can be filled with a *null*-value or omitted completely.

3.1.4 Example

See data/example/event.json

```
{
    "id": "550e8400-e29b-41d4-a716-446655440000",
    "timestamp": "2008-02-01T09:00:22+05:00",
    "type": "com.abb.ispf.event.echo",
    "belongsto": "550e8400-e29b-41d4-a716-446655440000",
    "payload": "BASE64 ENCODED STRING CONTAINING PAYLOAD DATA",
    "destination": []
}
```

3.2 Connection via Websocket over HTTPS

3.2.1 Websocket API endpoint

The websocket portal server will be available under the following URL path:

/socket

The complete URL for the websocket API endpoint will be

- wss://testing.ispf.desto.datadevelopment.de/socket
- or respectively wss://testing.ispf.debje.datadevelopment.de/socket

3.2.1.1 Processing order of an incoming websocket message

- The portal server will parse the incoming JSON messages
- The portal server will validate the supplied event type
- The portal server will verify if the supplied id is unique in its database
- The portal server will verify if the supplied belongsto id is available in its database

3.2.1.2 Acknowledgement for incoming websocket messages

In case of successful processing by the portal server

If the validation of the incoming message succeeded, the portal server will send a message as acknowledgement back to the portal client:

```
{
    "id": "7C65AA03-8AD7-4213-94A7-39B0CAB8714A",
    "timestamp": "2008-02-01T09:00:22+05:00",
    "type": "com.abb.ispf.event.success",
```

```
    "belongsto": "ID_OF_MESSAGE_FROM_PORTAL_CLIENT"
}
```

In case of an error

If the validation of the incoming message did not succeed, the portal server will send a message back to the portal client:

```
{
    "id": "C7E01BF5-7760-4E7A-B61A-7519EB7F3097",
    "timestamp": "2008-02-01T09:00:22+05:00",
    "type": "com.abb.ispf.event.error",
    "belongsto": "ID_OF_MESSAGE_FROM_PORTAL_CLIENT",
    "payload": "BASE64 ENCODED STRING WITH DETAILS ABOUT WHAT WENT WRONG"
}
```

The decoded Base64 payload data of the error event could have the following formings:

```
{
    "code": 409,
    "message": "Supplied id is not unique to the portal server"
}

{
    "code": 400,
    "message": "Supplied belongsto id is unknown to the portal server"
}

{
    "code": 400,
    "message": "Supplied type is unknown to the portal server"
}
```

3.3 Connection via HTTPS

3.3.1 HTTP API endpoint

The API for pushing events to the portal server will be available under the following URL path:

/api/event

3.3.1.1 The complete URL for the HTTP API endpoint will be

- wss://testing.ispf.desto.datadevelopment.de/api/event
- or respectively wss://testing.ispf.debje.datadevelopment.de/api/event

3.3.1.2 Accepted HTTP methods

- GET (request body will be ignored; will be specified later; used to query the portal server for events by criterias)
- POST (request body will be parsed; used to push new event to portal server)

If the portal client sends a request with a different HTTP method than the ones mentioned above, the portal server will respond with the HTTP status code 405 „Method not allowed“.

3.3.1.3 Pushing an event to the HTTP API endpoint

In order to push a new event to the portal server, the portal client must use the HTTP method „POST“. The portal client must send the generated JSON data as request body to the API. If the validation of the sent request body fails, the portal server will respond with the HTTP status code 400 "Bad Request". If the provided event id is not unique to the portal server, the portal server responds with the HTTP status code 409 "Conflict". The event must be send again with a new id.

If all went well, the portal server responds with the HTTP status code 201 "Created".

Summary of possible HTTP status codes when pushing to server

- 201 - Created; Everything went well
- 400 - Bad request; request body was malformed
- 403 - Forbidden; Certificate was not supplied to the portal server, certificate was revoked or not signed by the ISPf Root CA
- 405 - Method not allowed; Wrong HTTP method was used; only GET and PUT are allowed
- 409 - Conflict; an event with this id was already stored on the portal server
- 500 - Something along the processing on the portal server went wrong

Reasons for the returned HTTP status code 400 "Bad Request"

- Provided request body cannot be parsed (invalid JSON, …)

- Provided event type is unknown
- Provided belongsto id is unknown to the portal server

3.3.1.4 Fetching events from the HTTP API endpoint

Fetching a single event from the HTTP API endpoint

In order to fetch a single event from the portal server, the portal client must use the HTTP method „GET“. The ID of the requested event must be added to the HTTP API endpoint URL.

Example

/api/event/550e8400-e29b-41d4-a716-446655440000

Result


```
{
    "id": "550e8400-e29b-41d4-a716-446655440000",
    "timestamp": "2008-02-01T09:00:22+05:00",
    "timestamp_portal": "2008-02-01T09:00:22+05:00",
    "type": "com.abb.ispf.event.echo",
    "belongsto": "550e8400-e29b-41d4-a716-446655440000",
    "payload": "BASE64 ENCODED STRING CONTAINING PAYLOAD DATA",
    "destination": null
}
```


Fetching multiple events from the HTTP API endpoint

In order to fetch multiple events from the portal server, the portal client must use the HTTP method „GET“.

The API will support the following query parameters:

Type of event – type

Fetch only events of a specific type. A comma-separated list of different event types will also be supported.

```
type=com.abb.ispf.event.echo,com.abb.ispf.event.deviceinfo
```

ID of an event – id

Fetch only events with a specific ID. A comma-separated list of IDs will also be supported.

```
id=550e8400-e29b-41d4-a716-446655440000,C7E01BF5-7760-4E7A-B61A-7519EB7F3097
```

Newer than ID – newer_than_id

Fetch only events that are newer than the one with the ID specified.

```
newer_than_id=0A51AD34-0D8E-4835-915E-6AC7A984F331
```

Newer than point in time – newer_than

Fetch only events, that occured after the specified timestamp. The specified timestamp must be in ISO 8601 format.

```
newer_than=2008-02-01T09:00:22+05:00
```

Older than ID – older_than_id

Fetch only events that are older than the one with the ID specified.

```
older_than_id=6A87F1BE-C825-4931-904C-34CCBE00B693
```

Older than point in time – older_than

Fetch only events that are older than the specified timestamp. The specified timestamp must be in ISO 8601 format.

```
older_than=2013-09-10T16:00:22+01:00
```

Belonging to ID – belongsto

Fetch only events that are belonging to the event ID specified. A comma-separated list of IDs will also be supported.

```
belongsto=A2715FEF-7CBA-4A32-973C-DD25A4FC456E,8956AAD2-E231-4B74-AE6D-
B091EF37221E
```

Portal client – portal_client

Fetch only events that were send from the specified portal client (UUID of portal client). A comma-separated list of portal clients will also be supported.

```
portal_client=BF815D03-2619-4F10-AED1-58BC8FE6AB26,EEC490E4-0D88-4F41-B6E1-
6BEFD9A273C9,0E574ADB-8E39-49DC-A825-C0780351EF49
```

Pagination

Under some circumstances, the list of matching events can be very long. Therefore, the API will provide the possibility, to fetch only a smaller set of matching events.

The following query parameters can be used, to split the matching events into pages:

Number of returned events per page - pagination_limit (positive, non-zero integer)

Current page number - pagination_page (positive, non-zero integer)

Example

```
/api/event/?type=com.abb.ispf.event.echo&newer_than=2013-08-
31T00:00:00+01:00&pagination_limit=3&pagination_page=2
```

Result


```
{
    "count_total": 10,
    "events": [
        {
            "id": "DFBE0010-327E-49D4-BD3C-7D64A8D0143A",
            "timestamp": "2013-09-10T08:12:22+01:00",
            "timestamp_portal": "2013-09-10T08:12:22+01:00",
            "type": "com.abb.ispf.event.echo",
            "belongsto": null,
            "payload": "SGVsbG8gV29ybGQ=",
            "destination": null
        },
        {
            "id": "FA16130F-CA78-41A4-BB87-71123A4E5695",
            "timestamp": "2013-09-10T07:44:09+01:00",
            "timestamp_portal": "2013-09-10T07:44:09+01:00",
            "type": "com.abb.ispf.event.echo",
            "belongsto": null,
            "payload": "Zm9v",
            "destination": null
        },
        {
            "id": "7242410E-68FF-4B4E-8292-F7C8C56821F7",
            "timestamp": "2013-09-09T13:54:17+01:00",
            "timestamp_portal": "2013-09-09T13:54:17+01:00",
            "type": "com.abb.ispf.event.echo",
            "belongsto": null,
            "payload": "YmFy",
            "destination": null
        }
    ]
}
```


In this example, we requested the API to:

- Only return events of type „com.abb.ispf.event.echo“
- Only return events that are newer than 31th August 2013
- Only return three events per page
- Return the second page of matching events

The response body is supplemented by the total count of matching events. The matching events are embedded in the key „events“.