Examples for Handling the API

To facilitate handling of the application programming interface (API) of i-doit, we put together a few typical examples.

Contents

Login and Logout

Methods: **`idoit.login`**, **`idoit.version`** (as example for requests), **`idoit.logout`**

Request	Response

Header:
`X-RPC-Auth-Username: admin`
`X-RPC-Auth-Password: admin`
Body:
`{ "version": "2.0", "method": "idoit.login", "params": { "apikey": "xxx", "language": "en" }, "id": 1 }`
	Header:
	`X-RPC-Auth-Session: d1obs9m3d2pd8651grptjhdjg3`
	Body:
	`{ "jsonrpc": "2.0", "result": { "result": true, "userid": "9", "name": "i-doit system administrator ", "mail": "i-doit@acme-it.example", "username": "admin", "session-id": "d1obs9m3d2pd8651grptjhdjg3", "client-id": "1", "client-name": "ACME IT Solutions" }, "id": 1 }`

Header:
`X-RPC-Auth-Session: d1obs9m3d2pd8651grptjhdjg3`
Body:
`{ "version": "2.0", "method": "idoit.version", "params": { "apikey": "xxx", "language": "en" }, "id": 2 }`
	Header:
	`X-RPC-Auth-Session: d1obs9m3d2pd8651grptjhdjg3`
	Body:
	`{ "jsonrpc": "2.0", "result": { "login": { "userid": "9", "name": "i-doit system administrator ", "mail": "i-doit@acme-it.example", "username": "admin", "mandator": "ACME IT Solutions", "language": "de" }, "version": "1.9", "step": "", "type": "PRO" }, "id": 2 }`

Header:

```
X-RPC-Auth-Session: d1obs9m3d2pd8651grptjhdjg3
```

Body:

```
{
    "version": "2.0",
    "method": "idoit.logout",
    "params": {
        "apikey": "xxx",
        "language": "en"
    },
    "id": 3
}
```

Header:

```
X-RPC-Auth-Session: d1obs9m3d2pd8651grptjhdjg3
```

Body:

```
{
    "jsonrpc": "2.0",
    "result": {
        "message": "Logout successfull",
        "result": true
    },
    "id": 3
}
```

Creating a New Object

Method: **`cmdb.object.create`**

Request

Body:

```
{
    "version": "2.0",
    "method": "cmdb.object.create",
    "params": {
        "type": "C__OBJTYPE__SERVER",
        "title": "My little server",
        "apikey": "xxx",
        "language": "en"
    },
    "id": 1
}
```

Response

Body:

```
{
    "jsonrpc": "2.0",
    "result": {
        "id": "42",
        "message": "Object was successfully created",
        "success": true
    },
    "id": 1
}
```

Reading out General Information about an Object

Method: **`cmdb.object.read`**

Request	Response

Body:

```
{
    "version": "2.0",
    "method": "cmdb.object.
read",
    "params": {
        "id": 1000,
        "apikey": "xxx",
        "language": "de"
    },
    "id": 1
}
```

Body:


```
{
    "jsonrpc": "2.0",
    "result": {
        "id": "1000",
        "title": "ESXi1",
        "sysid": "VHOST_1426338622",
        "objecttype": "58",
        "type_title": "Virtual Host",
        "type_icon": "images\/icons\/silk\/server_database.png",
        "status": "2",
        "cmdb_status": "6",
        "cmdb_status_title": "In operation",
        "created": "2015-03-14 14:10:22",
        "updated": "2017-04-26 10:22:20",
        "image": "https:\/\/demo.i-doit.com\/images\/objecttypes\/server.
png"
    },
    "id": 1
}
```


Update of an Object

Method: **`cmdb.object.update`**

Request	Response

Body:

```
{
    "version": "2.0",
    "method": "cmdb.object.update",
    "params": {
        "id": 1000,
        "title": "esxi1",
        "apikey": "xxx",
        "language": "en"
    },
    "id": 1
}
```

Body:

```
{
    "jsonrpc": "2.0",
    "result": {
        "message": "Object title was successfully updated",
        "success": true
    },
    "id": 1
}
```

Archiving an Object/ Mark it as Deleted/ Purge

Method: **`cmdb.object.delete`**

Request	Response

Body:

```
{
    "version": "2.0",
    "method": "cmdb.object.delete",
    "params": {
        "id": 3240,
        "status": "C__RECORD_STATUS__ARCHIVED",
        "apikey": "xxx",
        "language": "en"
    },
    "id": 1
}
```

Body:

```
{
    "jsonrpc": "2.0",
    "result": {
        "message": "Object(s) successfully archived",
        "success": true
    },
    "id": 1
}
```

Body:

```
{
    "version": "2.0",
    "method": "cmdb.object.delete",
    "params": {
        "id": 3240,
        "status": "C__RECORD_STATUS__DELETED",
        "apikey": "xxx",
        "language": "en"
    },
    "id": 1
}
```

Body:

```
{
    "jsonrpc": "2.0",
    "result": {
        "message": "Object(s) successfully deleted",
        "success": true
    },
    "id": 1
}
```

Body:

```
{
    "version": "2.0",
    "method": "cmdb.object.delete",
    "params": {
        "id": 3240,
        "status": "C__RECORD_STATUS__PURGE",
        "apikey": "xxx",
        "language": "en"
    },
    "id": 1
}
```

Body:

```
{
    "jsonrpc": "2.0",
    "result": {
        "message": "Object(s) successfully purged",
        "success": true
    },
    "id": 1
}
```

Upload or Download of a Picture

Methods: **`cmdb.category.create`**, **`cmdb.category.read`**

Category: **`Pictures`**

Before uploading, you have to encode the file of the picture in BASE64. Also for downloading the picture has to be encoded in BASE64. For the sake of brevity, the strings encoded in BASE64 are replaced by wildcards in the examples.

Request	Response

Body:

```
{
    "version": "2.0",
    "method": "cmdb.category.create",
    "params": {
        "objID": 123,
        "data": {
            "name": "Picture of a cloud",
            "content": "<BASE64 encoded string>"
        },
        "catgID": "C__CATG__IMAGES",
        "apikey": "xxx",
        "language": "en"
    },
    "id": 1
}
```

Body:

```
{
    "jsonrpc": "2.0",
    "result": {
        "id": 7,
        "message": "Category entry successfully created.",
        "success": true
    },
    "id": 1
}
```

Body:

```
 {
    "version": "2.0",
    "method": "cmdb.category.read",
    "params": {
        "objID": 123,
        "category": "C__CATG__IMAGES",
        "apikey": "xxx",
        "language": "en"
    },
    "id": 2
}
```

Body:

```
 {
    "jsonrpc": "2.0",
    "result": [
        {
            "id": "7",
            "objID": "123",
            "name": "Picture of a cloud",
            "content": "<BASE64 encoded string>"
        }
    ],
    "id": 2
}
```

Upload a File and Assign It to an Object

Methods: **`cmdb.object.create`** and **`cmdb.category.create`**

Categories: **`Files`** **`File versions`** and **`Files`**

We want to upload the existing file **`test.txt`** to i-doit and assign it to a new server object. In i-doit files are objects, too. Before uploading the file its content have to be **BASE64** encoded, see attribute **`file_content`** in the 3rd request. As you can see the server object gets the ID **`1000`** and the file object the ID **`1001`**.

Step	Request	Response
Create server object	Body: `{ "version": "2.0", "method": "cmdb.object.create", "params": { "type": "C__OBJTYPE__SERVER", "title": "My little server", "apikey": "xxx", "language": "en" }, "id": 1 }`	Body: `{ "jsonrpc": "2.0", "result": { "id": 1000, "message": "Object was successfully created", "success": true }, "id": 1 }`

Create file object	Body:
	
	`{`
	`  "version": "2.0",`
	`  "method": "cmdb.object.create",`
	`  "params": {`
	`  "type": "C__OBJTYPE__FILE",`
	`  "title": "Just a test",`
	`  "apikey": "xxx",`
	`  "language": "en"`
	`  },`
	`  "id": 2`
	`}`
		Body:
		
		`{`
		`  "jsonrpc": "2.0",`
		`  "result": {`
		`  "id": 1001,`
		`  "message": "Object was successfully created",`
		`  "success": true`
		`  },`
		`  "id": 2`
		`}`

Upload file by category **`Files File versions`**	Body:
	
	`{`
	`  "version": "2.0",`
	`  "method": "cmdb.category.create",`
	`  "params": {`
	`  "objID": 1001,`
	`  "data": {`
	`  "file_content": "dGVzdAo=",`
	`  "file_physical": "test.txt",`
	`  "file_title": "Just a test",`
	`  "version_description": "Just a test"`
	`  },`
	`  "category": "C__CMDB__SUBCAT__FILE_VERSIONS",`
	`  "apikey": "xxx",`
	`  "language": "en"`
	`  },`
	`  "id": 3`
	`}`
		Body:
		
		`{`
		`  "jsonrpc": "2.0",`
		`  "result": {`
		`  "id": "69",`
		`  "message": "Category entry successfully created.",`
		`  "success": true`
		`  },`
		`  "id": 3`
		`}`

Assign file object to server object by category **`Files`**	Body:
	
	`{`
	`  "version": "2.0",`
	`  "method": "cmdb.category.create",`
	`  "params": {`
	`  "objID": 1000,`
	`  "data": {`
	`  "file": 1001`
	`  },`
	`  "category": "C__CATG__FILE",`
	`  "apikey": "xxx",`
	`  "language": "en"`
	`  },`
	`  "id": 4`
	`}`
		Body:
		
		`{`
		`  "jsonrpc": "2.0",`
		`  "result": {`
		`  "id": "69",`
		`  "message": "Category entry successfully created.",`
		`  "success": true`
		`  },`
		`  "id": 4`
		`}`

Installation of Software on Hardware

Method: **`cmdb.category.create`**

Category: **`Software allocation`**

Request	Response
Body: `{ "version": "2.0", "method": "cmdb.category.create", "params": { "objID": 123, "data": { "application": 456 }, "catgID": "C__CATG__APPLICATION", "apikey": "xxx", "language": "en" }, "id": 1 }` In this example, the hardware has the object ID 123 and the software has the object ID 456.	Body: `{ "jsonrpc": "2.0", "result": { "id": "500", "message": "Category entry successfully created.", "success": true }, "id": 1 }`

Document Model-specific Data of an Hardware Compoment

Method: **`cmdb.category.create`**

Category: **`Model`**

One specialty about this topic is that the attributes **`Manufacturer`** and **`Model`** are related to each other. In an API request you can set their names as strings because both are dialog+ fields. If one of these values does not exist it will be created automatically.

Request	Response
Body: `{ "version": "2.0", "method": "cmdb.category.create", "params": { "objID": 123, "data": { "manufacturer": "Name of manufacturer", "title": "Name of model" }, "category": "C__CATG__MODEL", "apikey": "xxx", "language": "en" }, "id": 1 }`	Body: `{ "jsonrpc": "2.0", "result": { "id": "183", "message": "Category entry successfully created.", "success": true }, "id": 1 }`