Documentation

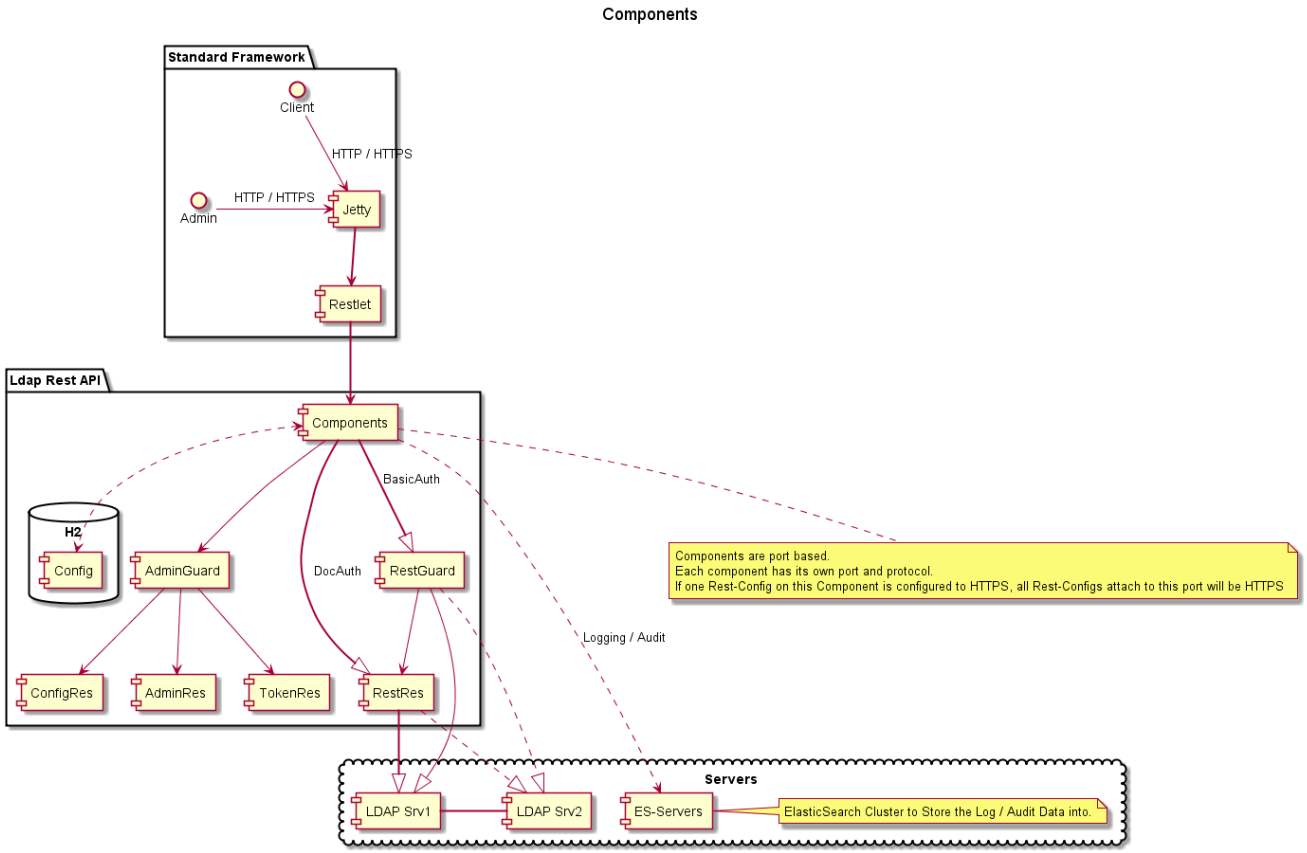
Installation and Configuration Guide

Ldap Rest API

Version:	1.1
created::	11.05.2017
File:	Ldap Rest API manual 1.docx

Contents

1 Service Design Overview



2 Installation

1 To install the Ldap Rest API, go to our web site http://www.skypro.ch/en/products and download the complete installation file *LdapRestAPI.zip*.
This file contains all the needed components. After downloading, please unpack it.

2 Extract the ZIP file into a new Directory

3 Create a Start.sh / start.cmd or start.command file.

The command to start the service is:

java -jar eDirRest-*-jar-with-dependencies.jar

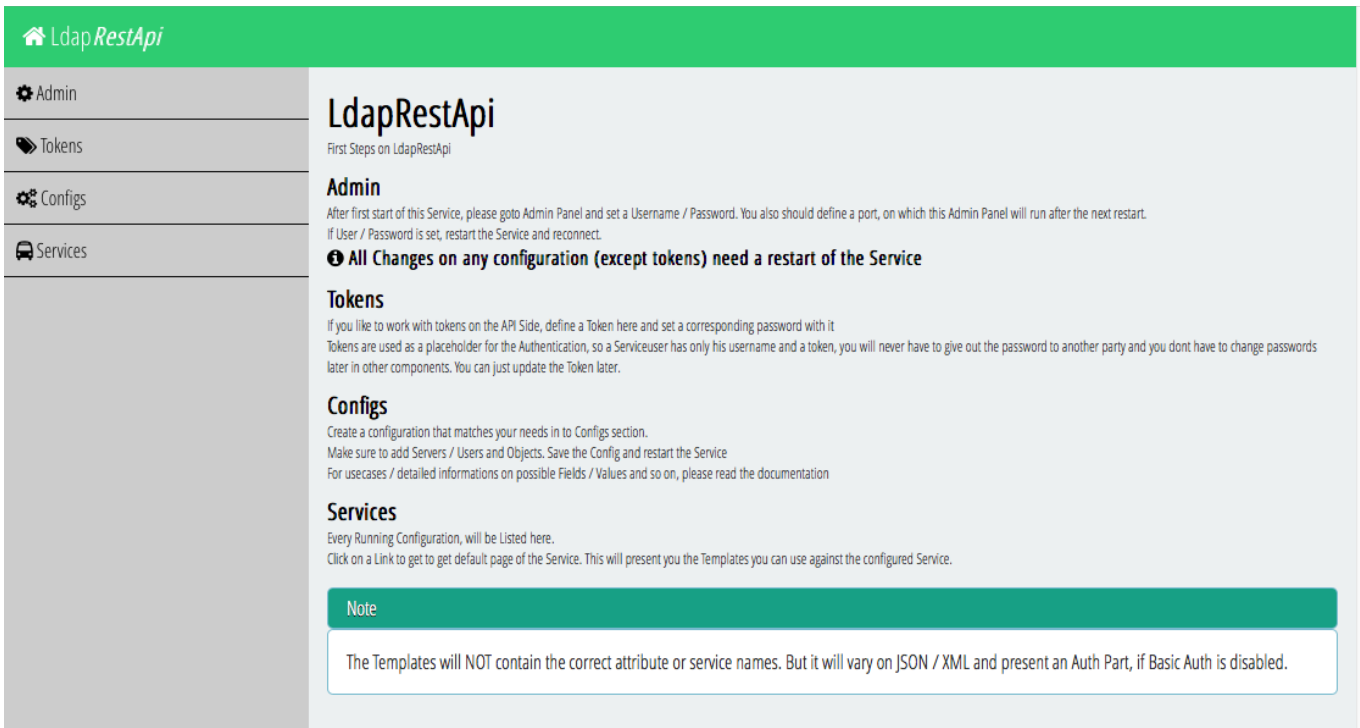
4 Start the service

Start the service via FireDaemon on Windows or any Wrapper available for you System.

5 First steps

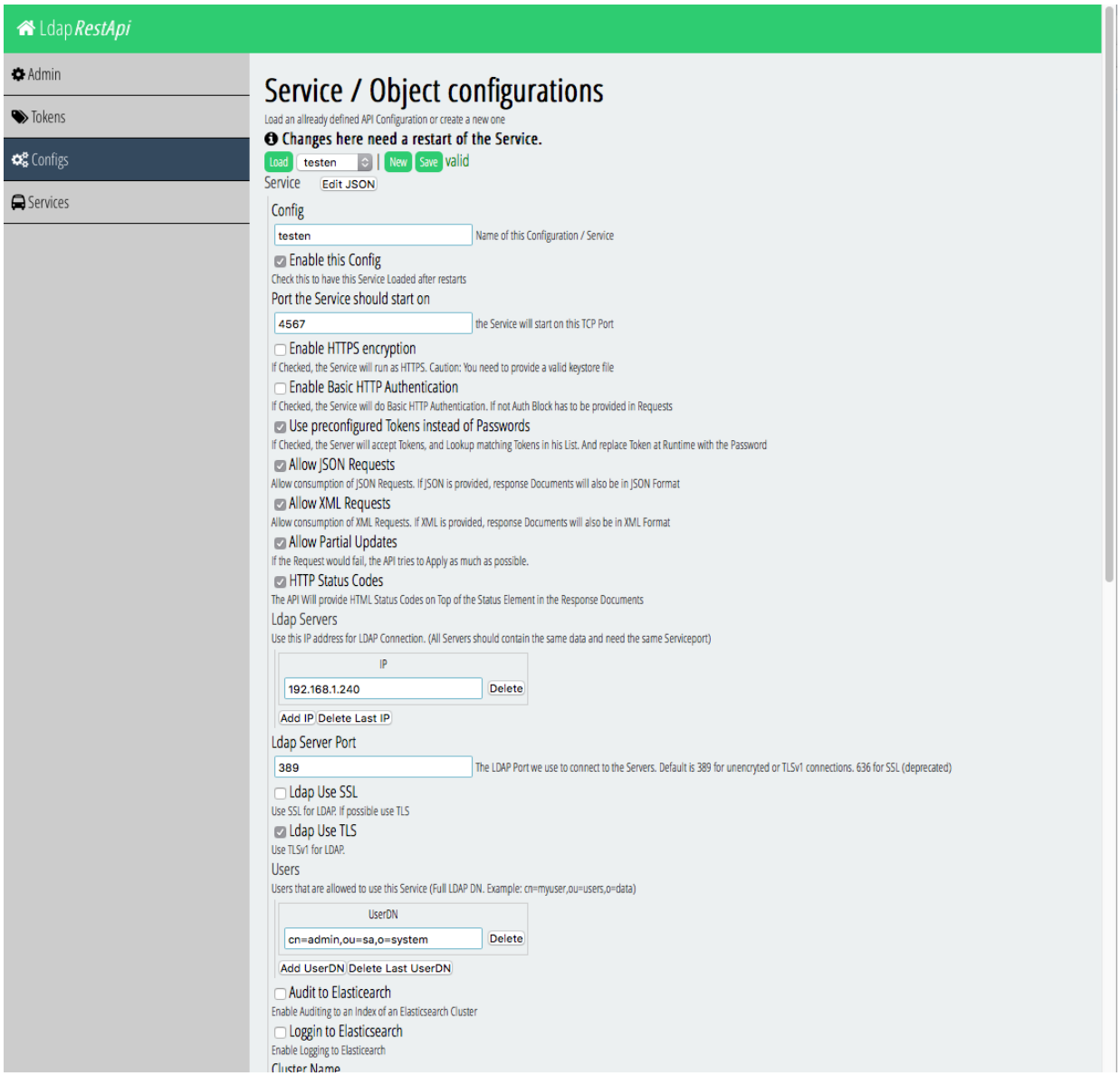
Connect to the Service via http://localhost:8182

You should be presented a Page like this:



Read the Page, and do as stated there.

The Basic Configuration may look like:



3 Startup Options / Recovery

If the Service does not start anymore, it may be due to the following two reasons.

1 One of the service has a Config, that does not start and is configured to run on the same port as the Adminserver
2 There is a keystore, that does not work with the given Configuration

To solve such problems, you can pass arguments to the service.

- You can move the AdminServer Port to an new unused port.
- You can disable HTTPS encryption for all the Configured Components / Services

Usage: java -jar LdapRestApi.jar

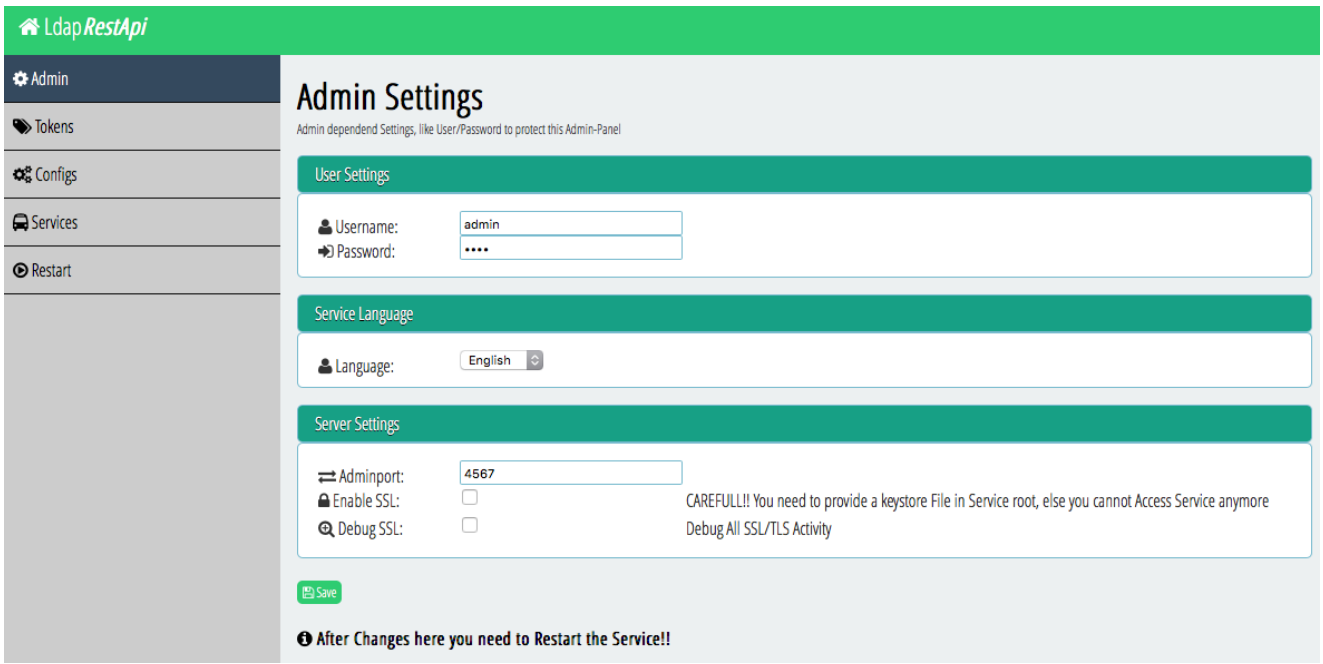
-h,--help print this message
-i,--insecure Force disable of HTTPS on the service
-p,--port <arg> Override port number of Admin Component

4 Configuration

Service configuration

After installation, you should be able to connect to http://localhost:8182

Go there to Admin and set Credentials for the Admin Gui of this service.



After you set the Adminport, Language, Username and Password you should restart the service and Connect to the new port.

You will be prompted for Username / Password, as soon as you access relevant Data APIs.

Note: Admin GUI / Rest Services may run on the same port. For simplicity and security, we suggest you run this on a different port.

Keystore for HTTPS encrytion

To create a Keystore file, you need the Java keytool or similar like OpenSSL.
The keystore file should be place in the root-directory of the Service (where the jar and db files are).

To create a Self-Signed keystore do:
keytool -keystore keystore -alias jetty -genkey -keyalg RSA

The Password for the keystore and the jetty server is "changeme" without ".

For more information check:

http://wiki.eclipse.org/Jetty/Howto/Configure_SSL

http://seppinho.github.io/restlet/webservice/2015/08/31/restlet/

Cacerts for LDAP

For the LDAP encryption we need a cacerts file, including the CA for the Certificate used by the LDAP.

Make sure, both (the CA and the Cert) Certificates are valid.

Add the CA by the following command:
keytool -importcert -trustcacerts -file cert.b64 -keystore cacerts -alias ldapservers

The cert.b64 is the CA Public Key. You do not need any private Keys.

Use "changeme" as Password for the cacerts file.

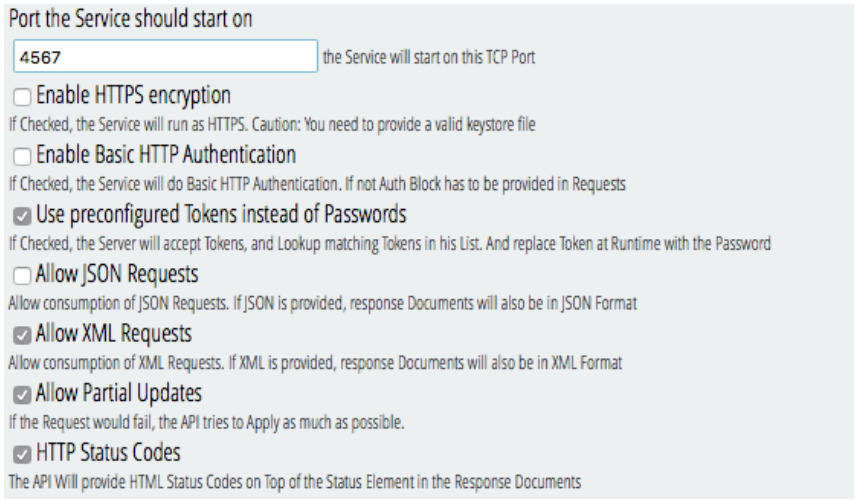
4.1 Configs

In the Configs section we configure Service / Object configurations. These are used to define the REST APIs, we present to the Use/Customer/Client.

A Config contains a Service Config which consists of:

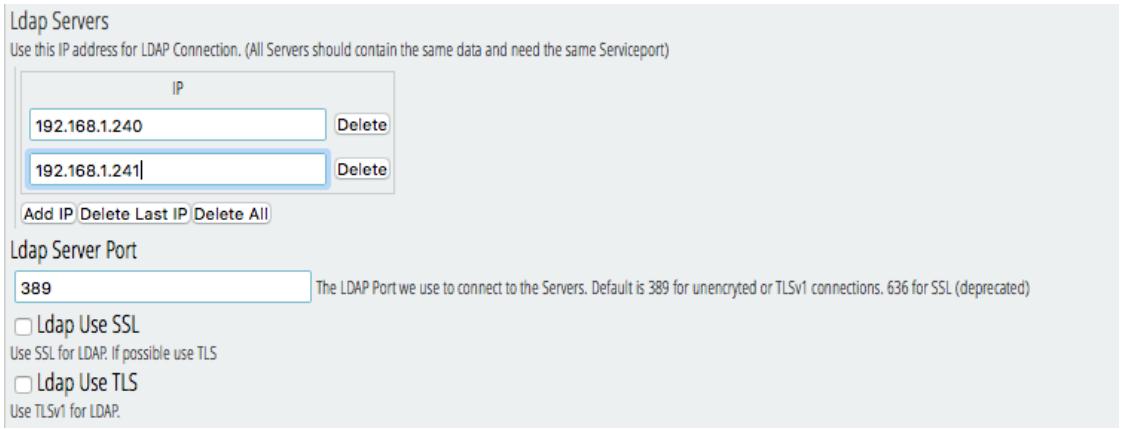
- Port / Security Config
- LDAP Servers for Auth and Datamanipulation
- Users
- Logging
- Object configuration
- Object specific settings
- Attributes

Port / Security Config



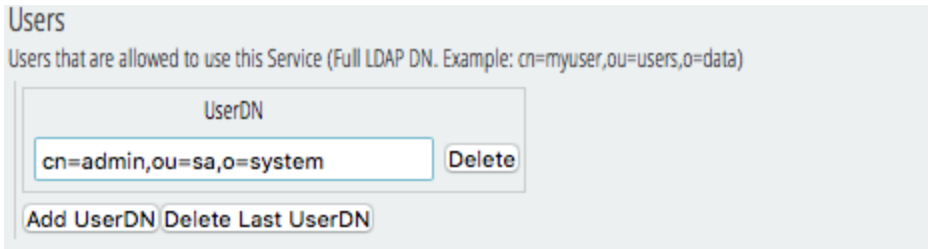
Name	Good Startvalue	Description
Port the Service should start on	4567	The Port this Configs runs on. All Configs with the same port, run on the same component. All Services on a component run HTTP or HTTPS.
Enable HTTPS encryption	false	Runt this Port as HTTPS? You need to create a keystore file in the service root folder. See documentation
Enable Basic HTTP Authentication	true/false	If checked, the Service requires Basic HTP Authentication. If not, you have to pass Auth parameters in the Request document
Use preconfigured Tokens instead of Passwords	true/false	If checked, you need to provide a token instead of a password. the service will replace the token with password stored in the token section.
Allow JSON Requests	true	If checked, you can pass requests as JSON to the service. You will receive a JSON file as response. Use Content-Type: application/json
Allow XML Requests	true	If checked, you can pass requests as XML to the service. You will receive a XML file as response. Use Content-Type: application/xml
Create more Generic Instance Output	false	Instead of simple XML / JSON, you can enable this Option to get a more Generic Document. In and Out
Allow Partial Updates	true/false	If checked, the Service tries to apply as many changes as possible. It will split up the operation to its atoms, and try to apply every value separatly. This happens only, if the Change fails as whole
HTTP Status Codes	true/false	If checked, the Status Codes provided in the answer documents will also be applied to the HTML Response.

LDAP Servers for Auth and Datamanipulation



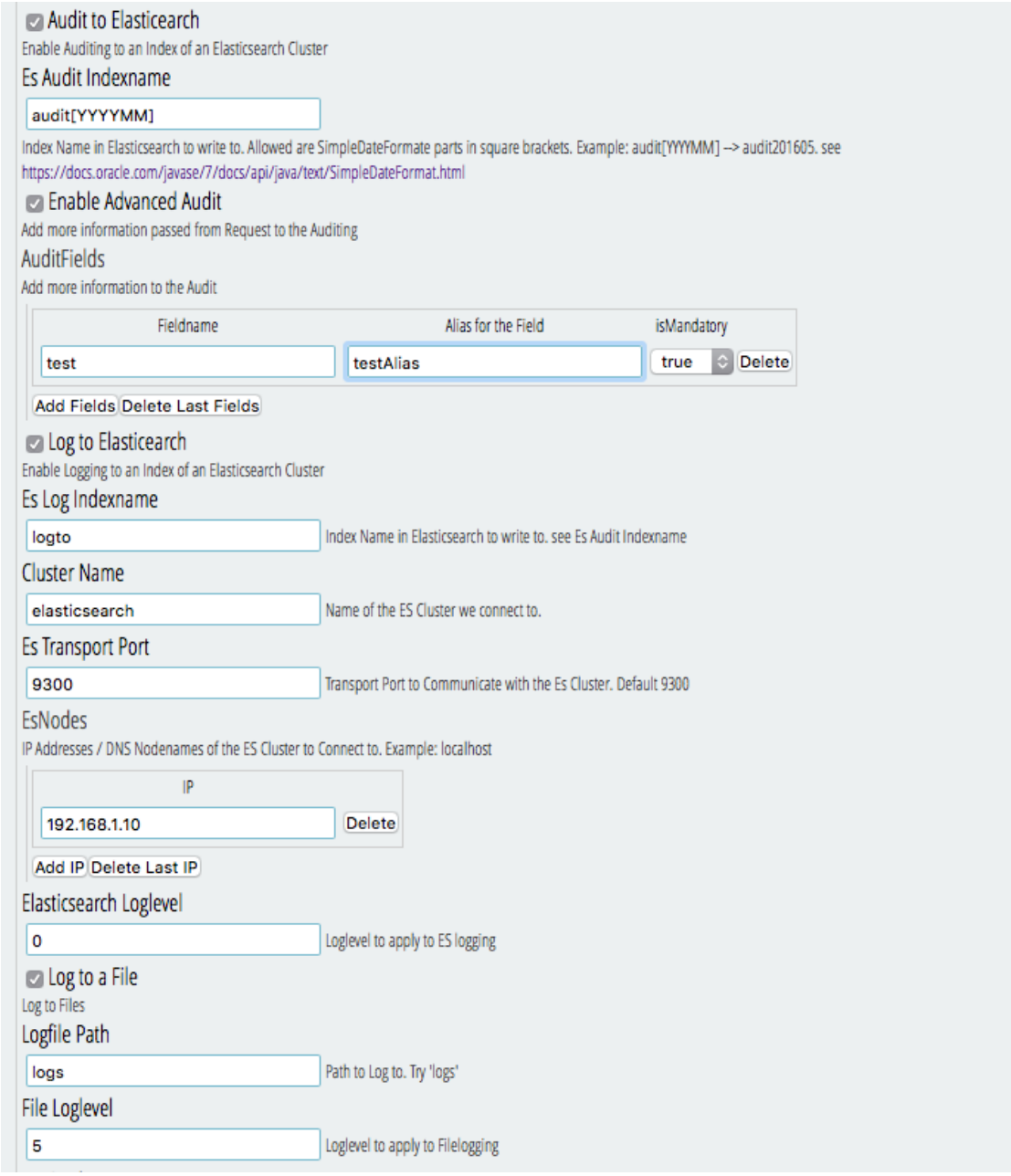
Name	Good Startvalue	Description
Ldap Servers	A List of your LDAP Servers	Add the IPs / DNS Names here of your LDAP server. The Rest Service will Auth against this Server and run its Operations against those Servers. It will allways use the first Server. If that is not available, it will the try to connect to the second one.
Ldap Server Port	389	The LDAP Port your Server responses to
Ldap Use SSL	false	Use SSL for LDAP. If possible use TLS
Ldap Use TLS	false	If checked, the Service tries to use TLSv1 for LDAP. (This runs on 389 as Extension / 636 is not required for security)

Users



Name	Good Startvalue	Description
UserDN	cn=admin,ou=sa,o=system	a fqdn of a user, you wanna give access to this API. NOTE: If no User / Users is provided, every available User of the LDAP Directory can use this API. This is Usefull, if you like to use this API in a Webpage or similar to change some personal Data like mobiles or whatever

Logging



Name	Good Startvalue	Description
Audit to Elasticsearch	true/false	If this box i checked, we try to log to an es cluster (v 1.x)
Es Audit Indexname	audit[YYYY]	The name of the Index we store our Audit Informations into. This Field can contain [] with the SimpleDateFormat to add dynamic to it. Examples: audit[YYYY] -> audit2016 audit[YYMM] -> audit1608

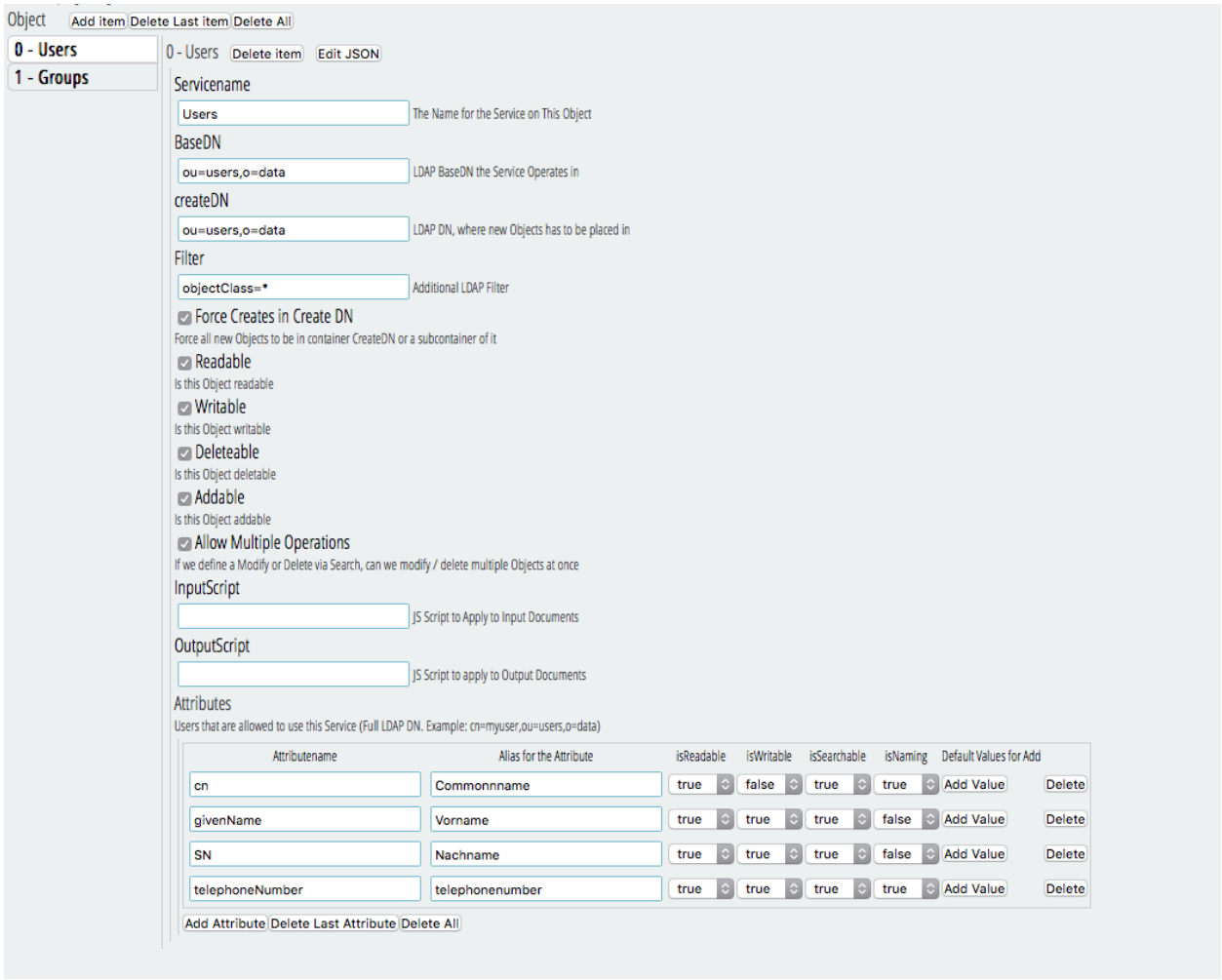
Enable Advanced Audit	true/false	If this is checked, we add Additional Information to the Auditobjects. See AuditFields

AuditFields	Requestor	If a Field is Added, we can have a Alias for it. That will be used in the Requestobject.
		If the Field is marked as Mandatory (isMandatory = true), the Service will Fail with Error 412, if no value is provided.
		
		The Content in the Request defined by:
		"Audit": [{
		"Requestor": {
		"value": "Hans Muster[USR007]"
		} will be stored in the Field "Requestor" with
		the Value "Hans Muster[USR007]"

Log to Elasticsearch	true/false	The Log from the Service will be sent to the ES Cluster
		The Entry looks like:
		"_source": {
		"message": "Check if User allowed for this Component: cn=admin,ou=sa,o=system",
		"postDate": "2016-0829T08:34:02.952Z",
		"data": null,
		"severity": "INFO",
		"severitycode": 3
		}

Es Log Indexname	log[YYYY]	The same as "Es Audit Indexname". Except this is for Logging purposes
Cluster Name	elasticsearch	The Clustername, the Transport client tries to connect to. The default is "elasticsearch"
Es Transport Port	9300	The Port the Transport client tries to reach the Cluster on (Transport not REST api !)
EsNodes	escluster	Ip or Hostname of the ClusterNodes. Need at least one valid, to get the Transport Client into the running Cluster. (Bootnodes)
Elasticsearch Loglevel	Warning	The Loglevel we use for ES Logging.
Log to File	true/false	Create Service Logfiles
Logfile Path	logs	Path to place the Logfiles into
File Loglevel	Info	The Loglevel we use for File Logging

Object configuration



Name	Good Startvalue	Description
Servicename	Users	This is the Servicename. The Configname will set the URI Part. This is the Reference you need to set in your Request <Service>
BaseDN	ou=users,o=data	This is the BaseDN used for all the Operations. You can leave it empty, for no restrictions. But we suggest, you enter a DN here
CreateDN	ou=users,o=data	This DN defines, in which Container / Subcontainer new Objects should be placed in.
Filter	objectClass=*	You need to provide a Filter here. Without (). It will restrict Search Operations / Multi Operation Finds

Allow Options to be used	false	If you want to change BaseDN / CreateDN or Filter on the fly through the Request enable this Option
Force Creates in Create DN	true	Use the Create DN for Adds instead of the BaseDN
Readable	true	This Objectconfig allows read. Needed for Search
Writeable	true	This Objectconfig allows write. Needed for Modifies
Deleteable	false	This Objectconfig allows delete. Needed for Deletes / Renames / Moves
Addable	true	Allows Adds. Also needed for Moves /Renames
Allow Multiple Operations	true	If this is checked. You can pass "Searches" for Modifies and Deletes, that will match multiple Objects. Operation is then performed on all finds If not checked, it will only performe an Operation, if only one Object has matched Search criteria.
InputScript/OutputScript		If this is provided, it will run this Script on the Object received/sent from/to a client. The scripts provided has to be Javascript. the simplest possible script would be: result=data; this will copy the data to the result. Which result in no change at all. For deeper insight. Check the Scripting documentation.

Attributes

Name	Good Startvalue	Description
Attributename		The name of the Attribute in the LDAP directory. Eg: CN / SN / givenName
Alias for the Attribute		The name this Attribute is presented in the REST API as. Can be left blank
isReadable	true	Attribute should be readable of the API
isWritable	true/false	Attribute can be written over the API
isSearchable	true/false	This Attribute can used as Searchattribute
isNaming	false	This attribute is a naming Attribute, can be used to auto generate DNs
Default Values for Add		Add default Values to this Attribute on add /create of Objects. For example create an attribute objectClass, set Readable / Writeable / Searchable / Nameing to false and set the Default values like Person / inetOrgPerson etc here.

4.2 Scripting

Scripts Folder

In the main folder of the Service, there is a scripts directory.
In that Directory there should be at least a helper.js Javascript file.
You can add as many js file to that directory as you like. They will get loaded when the Script Engine starts.

Script Invoke

To use a function of any of those scripts file, you simply include the method / function in the Config by Input / Output Script.

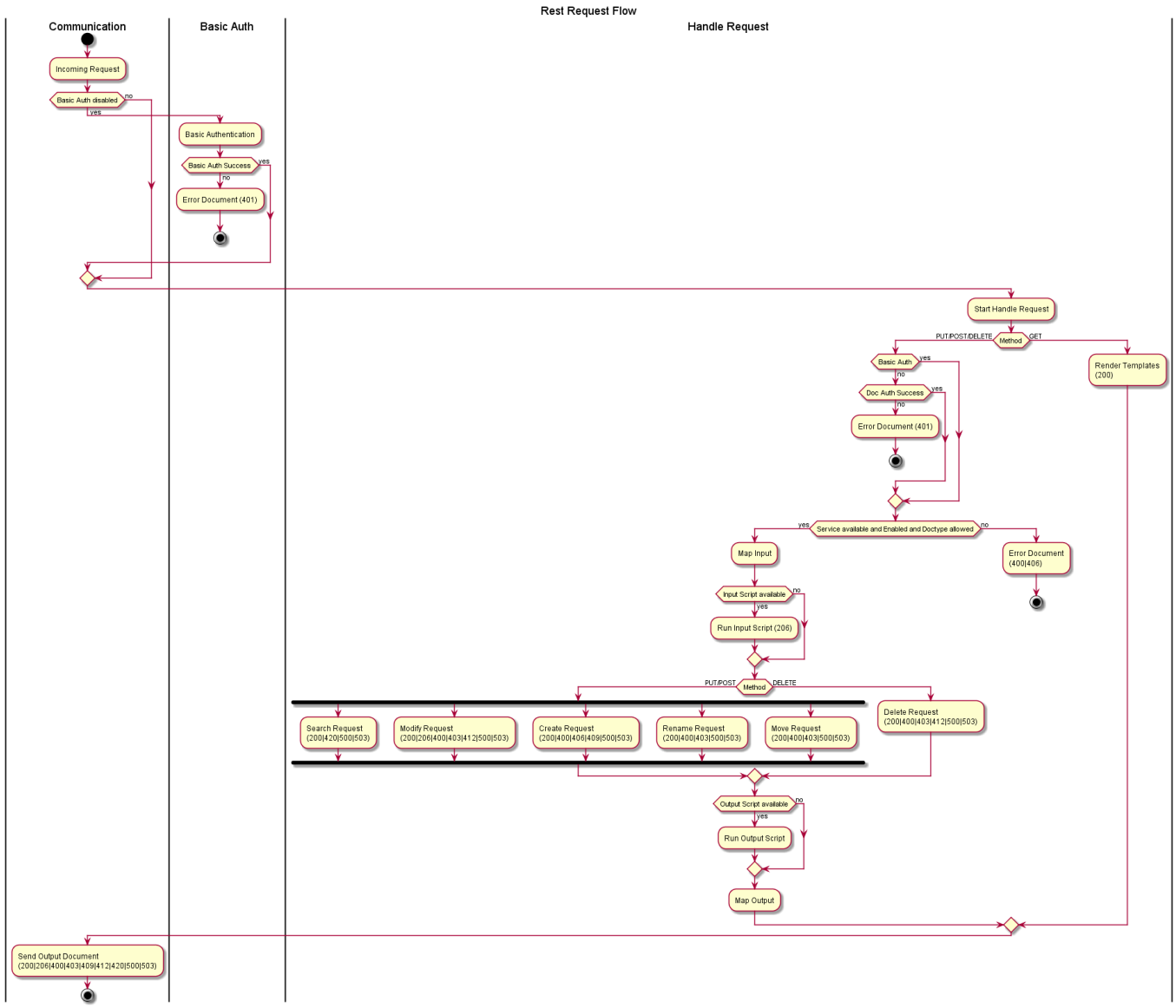
Get Started

Create your Hello World.

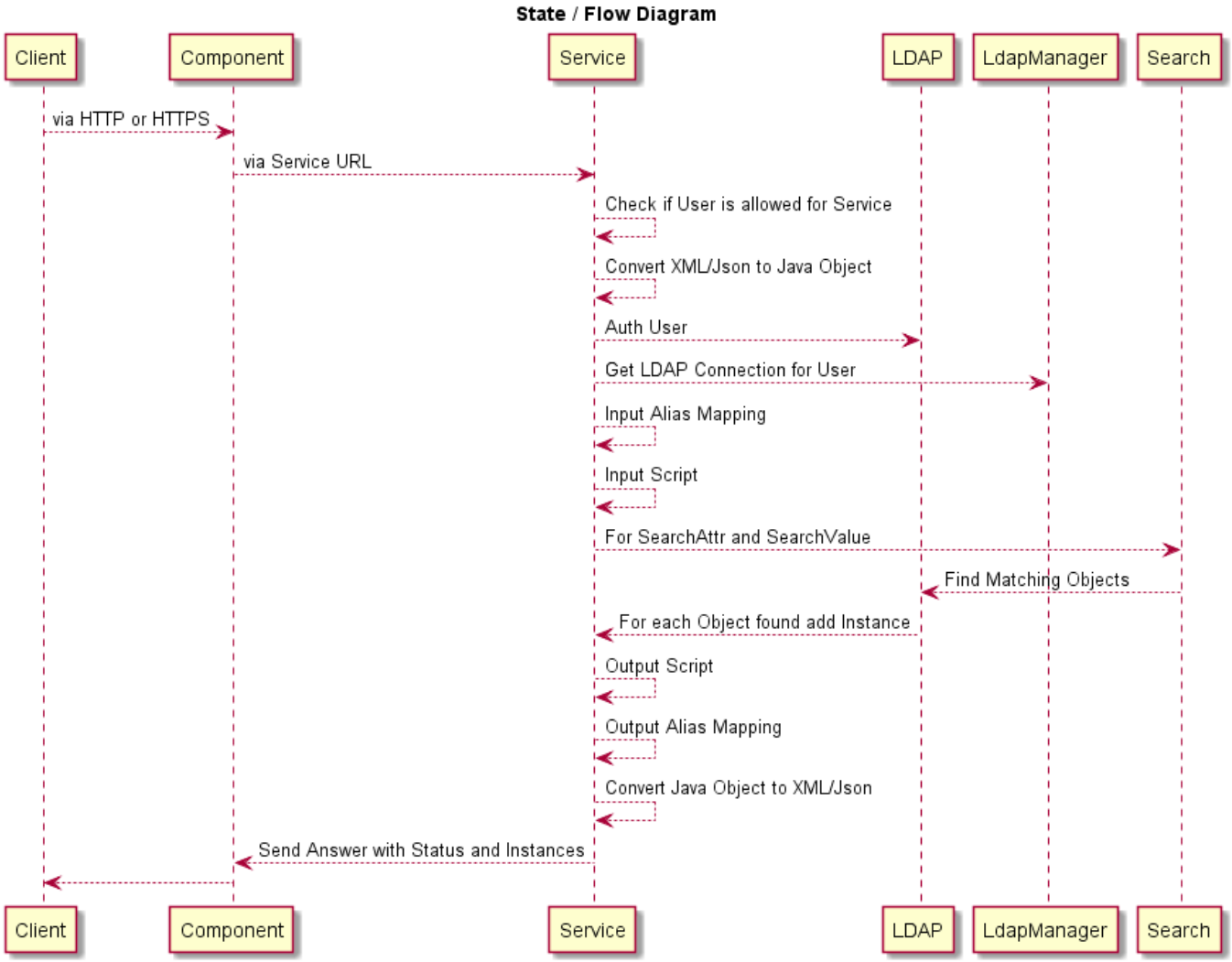
Paste the following Snipped into one of your Configs Input Script:
debug('hello World!!'); result=data;
This will print a line on the Server Console with 'hello World!!' and copy the input data to the output object. This means no changes are made to the data at all.
You can now start modify the Input data with normal Javascript and copy the needed Result to the result Object.

5 Service Flow Diagramme

General Request Flow



Search State / Flow



6 Error Codes / Handling

Error Codes are based on HTML Error Codes

https://de.wikipedia.org/wiki/HTTP-Statuscode

We use the following Error Codes:

Code	Message	Meaning
200	OK	All went as excepted
206	Partial OK	Some Operations could be performed (On Partial Operations enabled)
400	Bad Request	• The Request could not be processed • The Document could not be parsed
401	Unauthorized	Check your Credentials / Tokens
403	Forbidden	Not allowed by the Config
406	Unacceptable	Not a valid DN
409	Conflict	• Object Exists on Add • Does not Exist by Delete
412	Preconditions Failed	• Search / Filter matched more than one Object, but not enabled or allowed • Advanced Audit Value not provided
420	Method Failed	A given Attribute was not Found / is not allowed to Search on etc.
500	Server Error	Unexpected Error. Should not happen.
503	Ldap Error	No LDAP Connection available / All Servers down or unable to Connect

Error Document / Answers

The Response you receive, has an Status Section.
For Success it simply looks like (JSON)

```
"Status": {
"statuscode": 200,
"message": "OK"
}
```

Or as XML

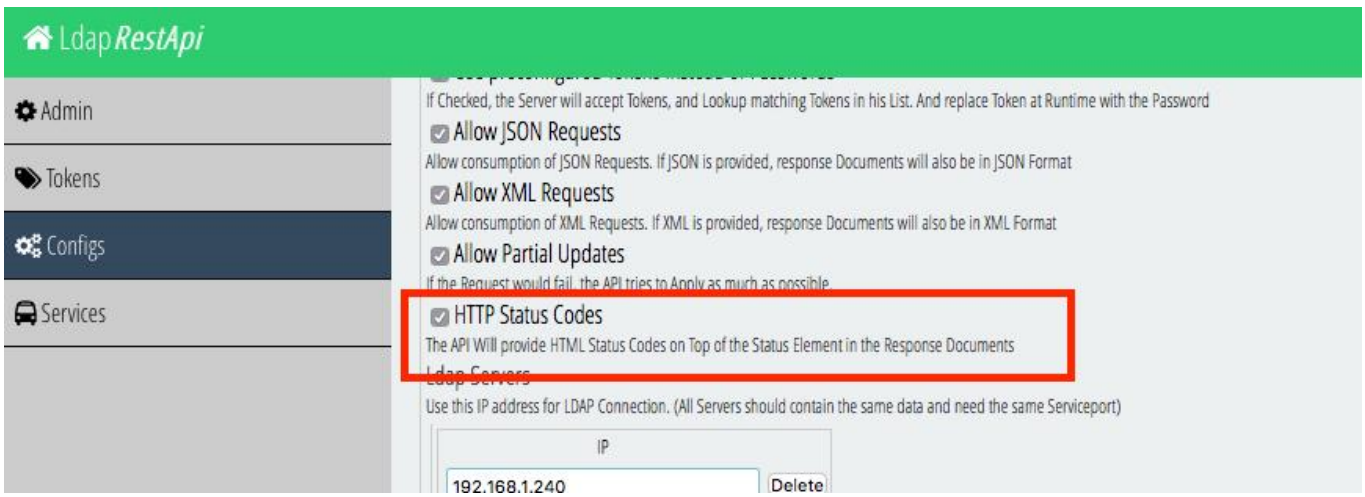
```
<Status>
<statuscode>200</statuscode>
<message>OK</message>
</Status>
```

If an Error happens,
this Section will contain the relevant Data. Also you may enable StackTrace to Doc in Config and end up with a Section like this:

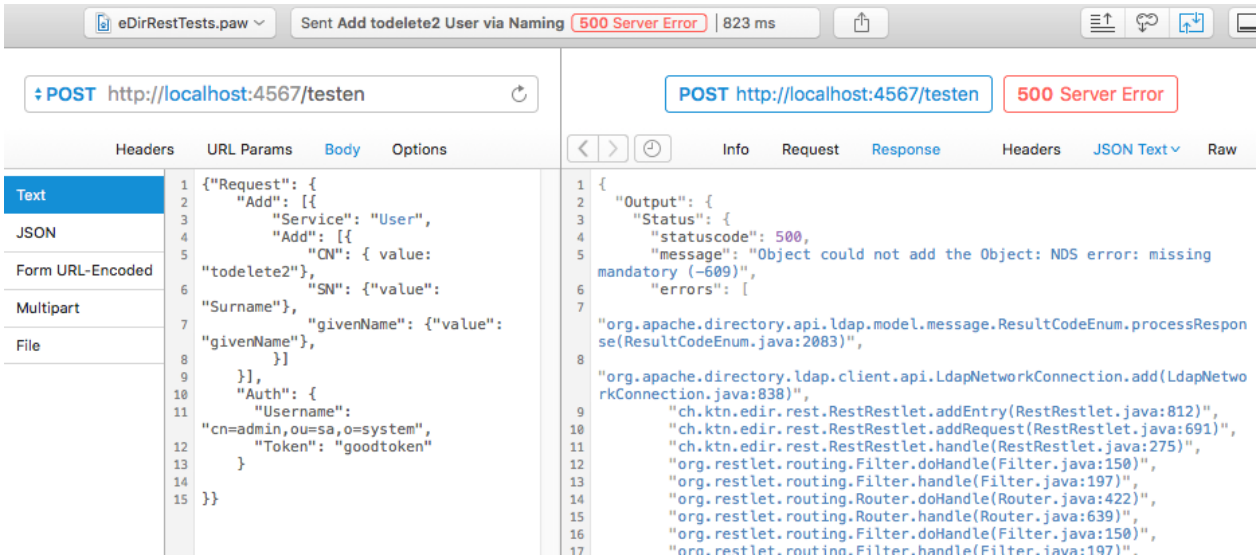
```
{
"Output": {
"Status": {
"statuscode": 409,
"message": "An Object with this DN allready exists. DN:
cn=todelete,ou=users,o=data -- Error: NDS error:
entry already exists (-606)",
"errors": [
"org.apache.directory.api.ldap.model.message.ResultCodeEnum.processRespons
e(ResultCodeEnum.java:2074)",
"org.apache.directory.ldap.client.api.LdapNetworkConnection.add(LdapNetwor
kConnection.java:838)",
"ch.ktn.edir.rest.RestRestlet.addEntry(RestRestlet.java:821)",
........
........
"java.lang.Thread.run(Thread.java:744)"
]
}
}
}
```

HTTP Status Codes

If you enable HTTP Status Codes



you get this Errors / Statuscodes also as HTTP Codes.



Defined Errors

Defined but not all are used at the moment.

Framework Errors

The underlaying Framework Restlet supports/depends on these Errors:
https://restlet.com/http-status-codes-map

Could not access the content at the URL because it is not from an allowed source.

https://restlet.com/http-status-codes-map

You may contact your site administrator and request that this URL be added to the list of allowed sources.

7 Request / Response Documents

Request as XML

Start Tag

XML Request

```
<Request>
```

Optional Options Block, if enabled in the Config

XML Request

```
<Options>
<BaseDN>ou=users,o=data</BaseDN>
<CreateDN>ou=users,o=data</CreateDN>
<Filter>objectClass=Person</Filter>
</Options>
```

One of the Following:

Search Block with Attr / Value

XML Request

```
<Search>
<SearchAtttribute>CN</SearchAtttribute>
<SearchValue>Value You Search for</SearchValue>
<Service>User</Service>
<ReadAttribute>givenName</ReadAttribute>
<ReadAttribute>eMail</ReadAttribute>
</Search>
```

Search Block with dn

XML Request

```
<Search>
<dn>uid=admin,ou=system</dn>
<Service>User</Service>
<ReadAttribute>givenName</ReadAttribute>
<ReadAttribute>eMail</ReadAttribute>
</Search>
```

Search Block with Attr / Value - Paged Search

XML Request

```
<Search>
<SearchAtttribute>CN</SearchAtttribute>
<SearchValue>Value You Search for</SearchValue>
<ObjectPerPage>10</ObjectPerPage>
<PagedSearch>true</PagedSearch>
<Service>User</Service>
<ReadAttribute>givenName</ReadAttribute>
<ReadAttribute>eMail</ReadAttribute>
</Search>
```

This will add a Cookie to the Response
<Output>.....<PagedCookie>TheCookie</PagedCookie></Output>
Search Block with Attr / Value - Paged Search – FollowUp

XML Request

```
<Search>
<SearchAtttribute>CN</SearchAtttribute>
<SearchValue>Value You Search for</SearchValue>
<ObjectPerPage>10</ObjectPerPage>
<PagedSearch>true</PagedSearch>
<PagedCookie>CookieFromLastRequest</PagedCookie>
<Service>User</Service>
<ReadAttribute>givenName</ReadAttribute>
<ReadAttribute>eMail</ReadAttribute>
</Search>
```

Modify Block

XML Request

```
<Modify>
<dn>DN of the Object to Modify or a Search</dn>
<SearchAtttribute>Attributename to search on if not using DN</SearchAtttribute>
<SearchValue>Value You Search for</SearchValue>
<Service>User</Service>
<ProcessMultiMatch>false</ProcessMultiMatch>
<Add>
<groups>
<value>group 1</value>
<value>group 2</value>
</groups>
</Add>
<Remove>
<groups>
<value>oldgroup 3</value>
</groups>
</Remove>
<Replace>
<givenName>
<value>new name</value>
<value>second new name</value>
</givenName>
</Replace>
</Modify>
```

Delete Block

XML Request

```
<Delete>
<dn>DN of the Object to Delete</dn>
<SearchAtttribute>Attributename to search on and delete finds</SearchAtttribute>
<SearchValue>Value you search for. Results will be Deleted</SearchValue>
<Service>User</Service>
<ProcessMultiMatch>false</ProcessMultiMatch>
</Delete>
```

Create Block

XML Request

```
<Create>
<dn>Enter the DN here or Class Default Container and Naming Attribute will be
used</dn>
<Service>User</Service>
<Add>
<cn>
<value>cn - for dn creation if this is checked as naming</value>
</cn>
<SN>
<value>Surname</value>
</SN>
<givenName>
<value>givenName</value>
</givenName>
</Add>
</Create>
```

Rename Block

XML Request

```
<Rename>
<OldDN>cn=oldname,o=demo</OldDN>
<Newname>cn=newname</Newname>
<Service>User</Service>
</Rename>
```

Move Block

XML Request

```
<Move>
<OldDN>cn=oldname,o=demo</OldDN>
<TargetContainer>ou=target,o=newname</TargetContainer>
<Service>User</Service>
</Move>
```

Optional: Auth Block

XML Request

```
<Auth>
<Username>cn=myuser,ou=technicalusers=o=unit</Username>
<Token>Token because Token Authentication is Enabled</Token>
</Auth>
```

XML Request

```
<Auth>
<Username>cn=myuser,ou=technicalusers=o=unit</Username>
<Password>MySecret</Password>
</Auth>
```

Optional: Audit Block

XML Request

```
<Audit>
<advAuditField>
<value>If AdvAudit is enabled, you pass values like this</value>
</advAuditField>
</Audit>
```

Stop Tag

XML Request

```
</Request>
```

A Full example would look like:

XML Request

```
<Request>
<Search>
<SearchAtttribute>CN</SearchAtttribute>
<SearchValue>Value You Search for</SearchValue>
<Service>User</Service>
<ReadAttribute>givenName</ReadAttribute>
<ReadAttribute>eMail</ReadAttribute>
</Search>
<Auth>
<Username>cn=myuser,ou=technicalusers=o=unit</Username>
<Token>Token because Token Authentication is Enabled</Token>
</Auth>
</Request>
```

The structure is the same in JSON. As an example:

JSON Request

```
{"Request": {
"Modify": {
"dn": "DN of the Object to Modify or a Search",
"SearchAtttribute": "Attributename to search on if not using DN",
"SearchValue": "Value You Search for",
"Service": "User",
"ProcessMultiMatch": false,
"Add": [{"groups": {"value": [
"group 1",
"group 2"
]}}],
"Remove": [{"groups": {"value": "oldgroup 3"}}],
"Replace": [{"givenName": {"value": [
"new name",
"second new name"
]}}]
},
"Auth": {
"Username": "cn=myuser,ou=technicalusers=o=unit",
"Token": "Token because Token Authentication is Enabled"
}
}}
```

Response Documents

A normal response document in XML would be:

XML Response

```
<Output>
<Data>
<Object>
<dn>cn=anObject,ou=anOrgUnit,o=Org</dn>
<Content>
<cn>
<value>anObject</value>
</cn>
<givenName>
<value>My Name</value>
<value>My Other Name</value>
</givenName>
</Content>
</Object>
</Data>
<Status>
<statuscode>200</statuscode>
<message>OK</message>
</Status>
</Output>
```

If an Error happen, you get the Error in the Status section. It looks like this:

XML Response

```
<?xml version="1.0"?>
<Output>
<Data/>
<Status>
<statuscode>420</statuscode>
<message>The given Attribut is not searchable. Check your config.</message>
</Status>
</Output>
```

If you enable Generic Instance Output, you will get a more Generic Document for Input and Output. If it is checked, you can still send simple Documents to the API

Example of a more Generic Doc:

XML Response

```
<Request>
<Modify>
<dn>DN of the Object to Modify or a Search</dn>
<SearchAtttribute>Attributename to search on if not using DN</SearchAtttribute>
<SearchValue>Value You Search for</SearchValue>
<Service>User</Service>
<ProcessMultiMatch>false</ProcessMultiMatch>
<Add>
<Attribute>
<name>groups</name>
<value>group 1</value>
<value>group 2</value>
</Attribute>
</Add>
<Remove>
<Attribute>
<name>groups</name>
<value>oldgroup 3</value>
</Attribute>
</Remove>
<Replace>
<Attribute>
<name>givenName</name>
<value>new name</value>
<value>second new name</value>
</Attribute>
</Replace>
</Modify>
<Auth>
<Username>cn=myuser,ou=technicalusers=o=unit</Username>
<Token>Token because Token Authentication is Enabled</Token>
</Auth>
<Audit>
<Attribute>
<name>advAuditField</name>
<value>If AdvAudit is enabled, you pass values like this</value>
<value>If an Alias is defined</value>
</Attribute>
</Audit>
</Request>
```