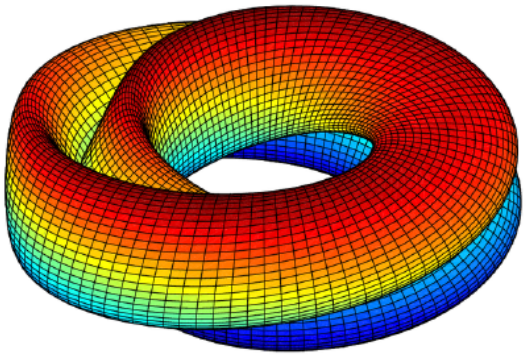
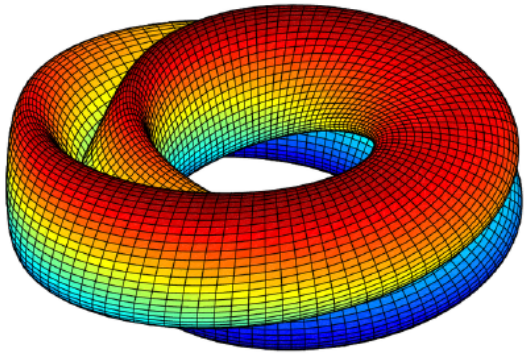
Skylar "die Gelehrte"



Thomas Lutz



Skylar "die Gelehrte":

Thomas Lutz

Table of Contents

Installation

Erstinstallation

Installieren Sie Docker [https://docs.docker.com/engine/installation/], docker-compose [https://docs.docker.com/compose/install/] und Git [https://git-scm.com/downloads].

```
# Hinterlasse eine Nachricht um den source code zu erhalten
# https://github.com/phoen1x/skylar-the-scholar/issues

# build and start skylar
cd skylar-the-scholar
./container_start.sh

# wait for skylar-core to start up
# Grab some coffee this will take some time...
```

Öffne http://skylar.livingfire.de in Deinem Webbrowser.

Upgrade

Lies das Changelog und erstelle ein Backup deiner Daten.

```
# stop skylar
PROJECT="$HOME/git/skylar"
cd $PROJECT
./container_destroy.sh

# upgrade
git reset --hard
git pull

# build and start Skylar
./container_start.sh build
```

Uninstall

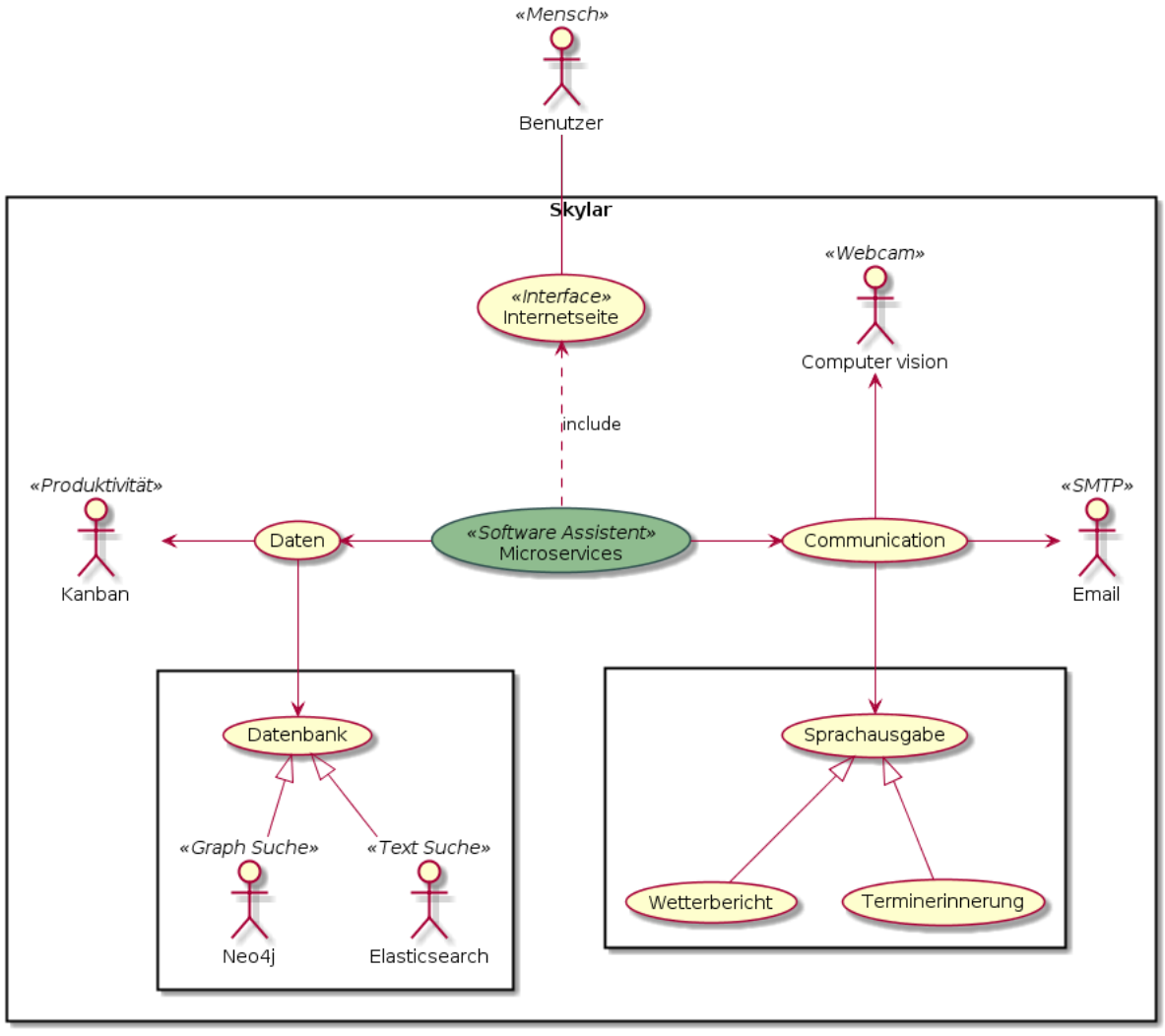
```
# stop skylar
PROJECT="$HOME/git/skylar"
cd $PROJECT
./container_destroy.sh

# delete the skylar folder
cd ..
rm -rf skylar
```

Technik

Docker containers - technik

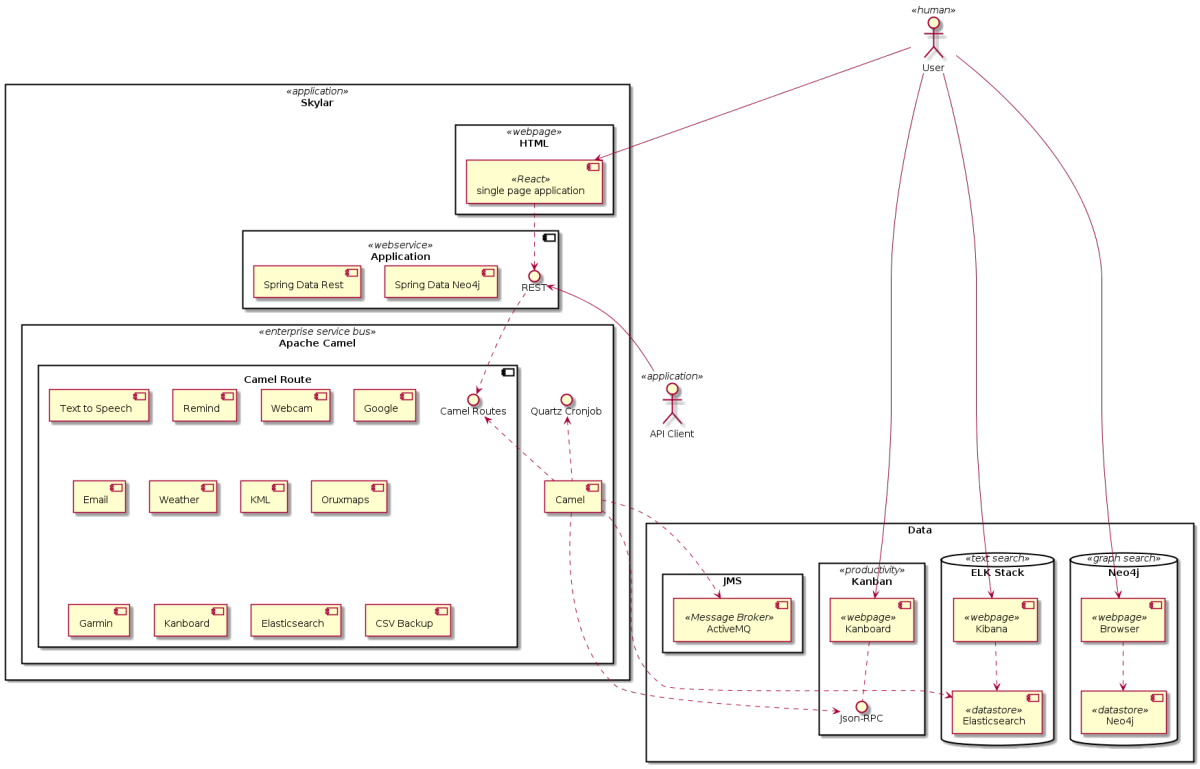
```
PROJECT="$HOME/git/skylar"
cd $PROJECT

# Build Java jar files and start application
./container_start.sh build

# start application
./container_start.sh

# delete docker container
./container_destroy.sh
```

Uebersicht - technik



[media/book_technical_overview_de.png]

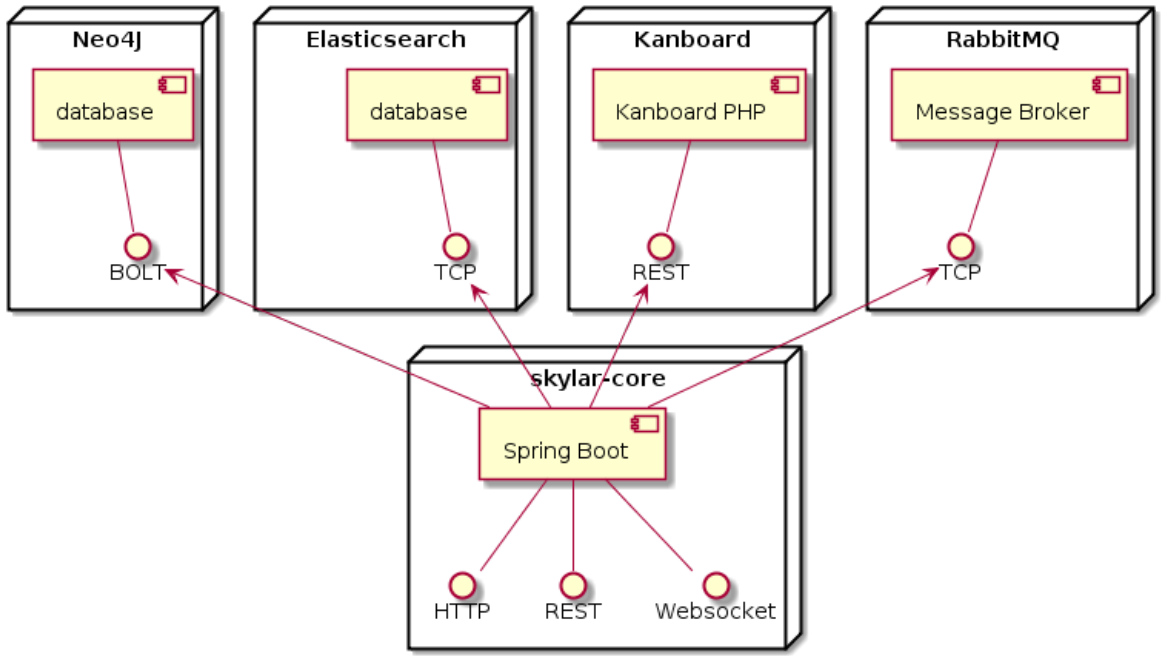
Details - technik



[media/book_technical_inner_workings.png]

skylar-core

Abhaengigkeiten - skylar-core



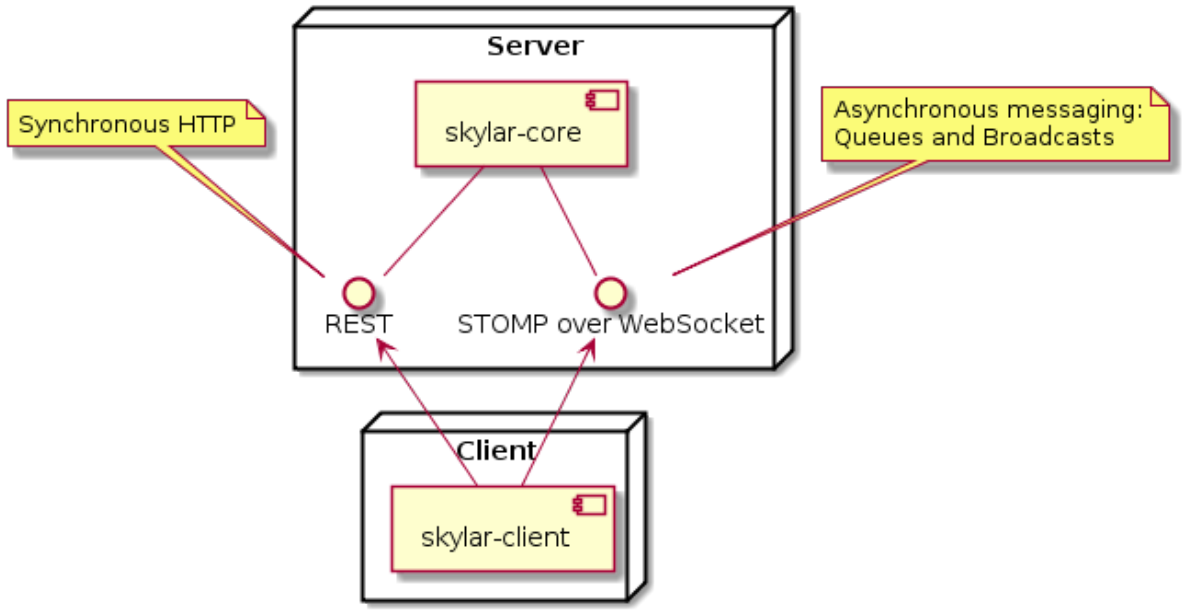
[media/book_skylar_core_overview.png]

API - skylar-core

Restful API Dokumentation [http://www.livingfire.de/skylar-the-scholar/api/skylar-core]

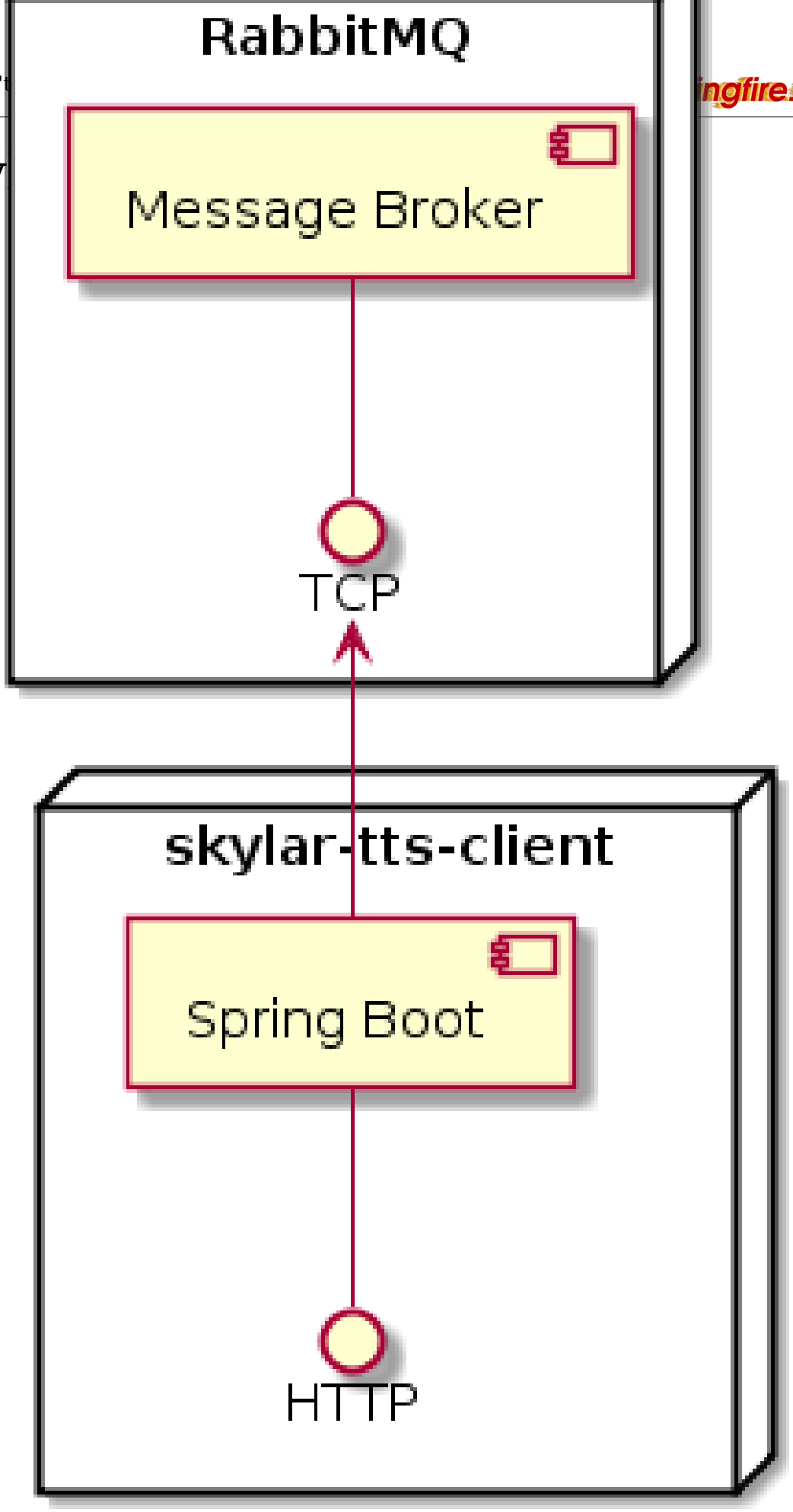
skylar-client

Abhaengigkeiten - skylar-client



[media/book_skylar_client_dependencies.png]

sky



[media/book_skylar_tts_client_overview.png]

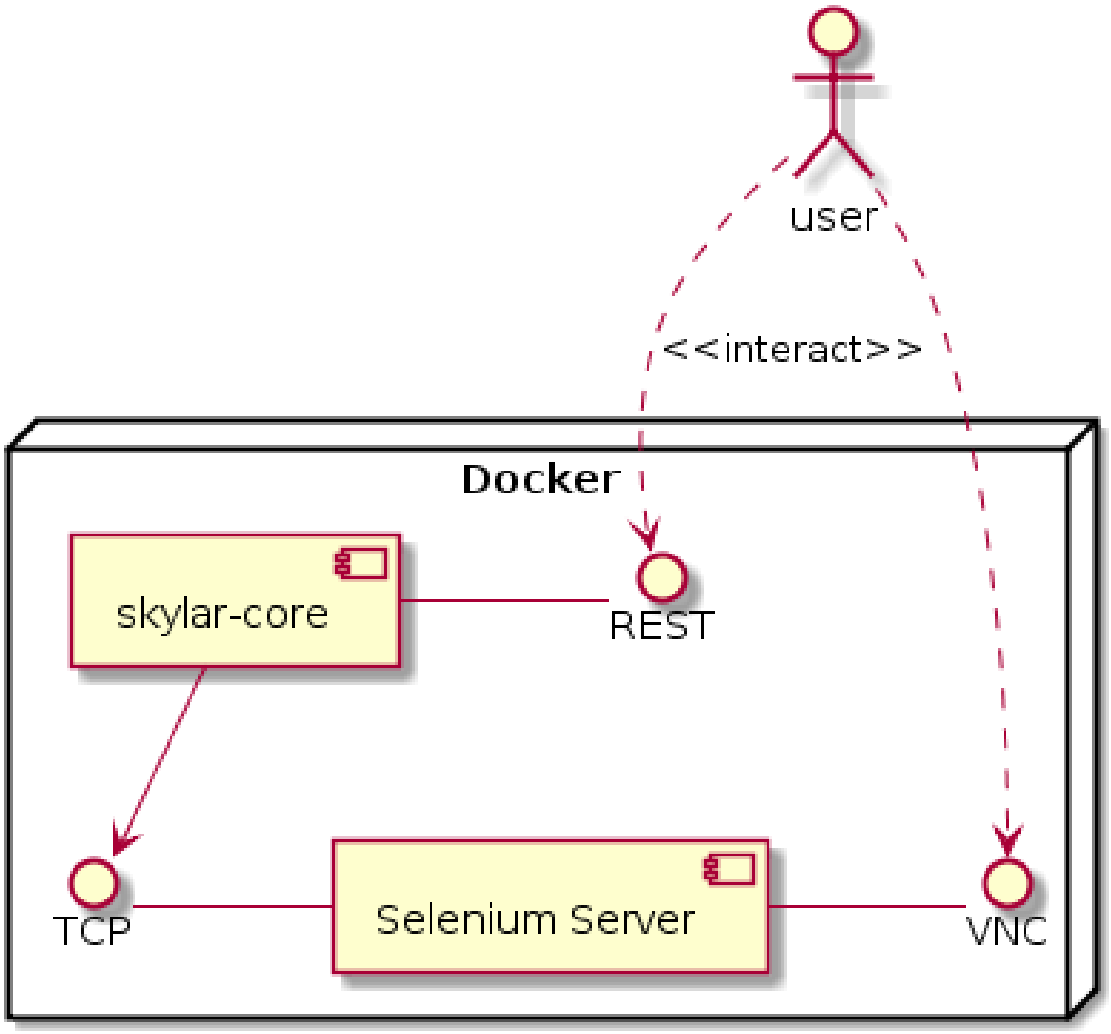
Gebrauch - skylar-tts-client

Mit diesem Client kann man die Sprachausgabe von Skylar in einen weiteren Raum oder ein anderes Stockwerk des Hauses übertragen.

```
PROJECT="$HOME/git/skylar"

# make sure the TCP port in RABBITMQ
# is available to external clients
atom $PROJECT/docker/docker-compose.yml

# start headless tts client
cd $PROJECT/docker/_volumes/skylar/skylar-tts-client
java -jar app.jar
```

Selenium

Abhaengigkeiten - selenium



[media/book_skylar_selenium_overview.png]

Bedienung - selenium

Siehe REST API section Selenium [http://www.livingfire.de/skylar-the-scholar/api/skylar-core/#api-skylar-core-api-selenium] wie man Skylars Firefox fernsteuert. Um den Intnernetbrowser zu sehen verwendet man einen VNC viewer [https://www.realvnc.com/de/connect/download/viewer/].

```
# Open your favorite VNC viewer
# https://www.realvnc.com/en/connect/download/viewer/
#     host: 127.0.0.1
#     port: 61622
#     password: skylar
xdg-open 'vnc://:skylar@127.0.0.1:61622'
# Now you should see a black screen with a Ubuntu logo
```

```
# more info https://github.com/SeleniumHQ/docker-selenium
```

Einschränkungen des Docker container - selenium

Früher oder später benötigt man Ton oder Mikrophone für Skylars Internetbrowser. Dies erreicht man am leichtesten wenn man einen Selenium Server in der Desktop Umgebung also außerhalb von Docker startet. Eine weitere Möglichkeit ist das aktivieren der benötigten Funktionen in Deinem Linux/Mac/Windows/ARM Docker Deamon. Jedoch hat jedes Betriebssystem seine eigene Einstellungen so wie Heimtücken...

Konfiguration des externen Selenium Server - selenium

```
PROJECT="$HOME/git/skylar"

# stop application
cd $PROJECT
./container_destroy.sh

# change selenium server to 172.23.42.1:4444
cd $PROJECT/docker/_volumes/registry
sed -i -e 's|selenium:4444/wd/hub|172.23.42.1:4444/wd/hub|' core.yml

# start application
cd $PROJECT
./container_start.sh
```

Start des externen Selenium Server - selenium

```
# Start selenium server
PROJECT="$HOME/git/skylar"
cd $PROJECT/tools/selenium
./startServer.sh
```

Weitere Inforamtionen finden sich in meinem Selenium Artikel [http://www.livingfire.de/skylar/software-assistent-spring-rest-selenium-en/]

Ton

Methode 1: Gesamte Soundkarte - selenium

```
# edit docker-compose.yml
PROJECT="$HOME/git/skylar"
atom $PROJECT/docker/docker-compose.yml
```

diese Zeilen auskommentieren

```
selenium:
    volumes:
    #            - "$DEVICE_SHM"
```

```
    #            - "$VOLUME_PULSEAUDIO_DBUS"
    #            - "$VOLUME_PULSEAUDIO_MACHINE_ID"

    #        devices:
    #            - "$DEVICE_SOUND"
```

Restart pulseaudio - selenium

Beim aktiveren des Tons wird Docker die Soundkarte sperren und es kein Sound mehr von der Desktop Umgebung abgespielt werden.

```
# stop docker
PROJECT="$HOME/git/skylar"
cd $PROJECT/docker
docker-compose down

# make sure all containers are down
docker ps -a | grep skylar

# restart pulseaudio
pulseaudio -k && sudo alsa force-reload
```

Methode 2: Stream PulseAudio mit TCP - selenium

Z.B. günstiger Raspberry Pi mit Ubuntu [https://www.raspberrypi.org/downloads/]

Setup TCP Server

```
sudo -i
# install pulseaudio
apt-get -y install \
                alsa-utils \
                libasound2 \
                libasound2-plugins \
                pulseaudio \
                pulseaudio-module-zeroconf \
                pulseaudio-utils \
                --no-install-recommends

# backup configuration
cp /etc/pulse/default.pa /etc/pulse/default.pa.org

# add TCP configuration --- replace network 192.168.0.0/16
echo '
# tcp server
load-module module-native-protocol-tcp auth-ip-acl=127.0.0.1;192.168.0.0/16 auth-anonymous=1
load-module module-zeroconf-publish
' >> /etc/pulse/default.pa

# restart linux box
reboot
```

Test TCP Server

```
# make shure you are NOT ROOT and are in AUDIO GROUP
id | grep -v root | grep audio

# Add your user to the audio group if needed then restart shell
# sudo usermod -aG audio,pulse,pulse-access USER
# exit

# test audio
# locate wav | grep wav$
aplay /usr/share/sounds/alsa/Noise.wav

# check tcp module loaded
pactl list | grep module-native-protocol-tcp

# start server in debug mode
pulseaudio -v

# use SECOND TERMINAL to test --- replace IP 192.168.0.10
PULSE_SERVER=192.168.0.10:4713 pavucontrol
```

```
# play tcp sound --- replace IP 192.168.0.10
PULSE_SERVER=192.168.0.10:4713 vlc

# more info
xdg-open https://wiki.ubuntuusers.de/PulseAudio/#Soundausgabe-im-Netzwerk-umleiten
```

Run PulseAudio as systemd service

```
sudo -i

# create service script
echo '[Unit]
Description=System PulseAudio sound server

[Service]
Type=notify
ExecStart=/usr/bin/pulseaudio --verbose --system --daemonize=no --high-priority --log-target=syslog --disallow-exit --disallow-module-

[Install]
WantedBy=multi-user.target' > /etc/systemd/system/pulseaudio.service

# enable and start daemon
systemctl enable pulseaudio
systemctl start pulseaudio
```

Setup Docker

Ändere den .env Wert ENV_PULSE_AUDIO_TCP in der Datei $PROJECT/docker/.env

```
ENV_PULSE_AUDIO_TCP=PULSE_SERVER=192.168.0.10:4713
```

```
# edit .env file
PROJECT="$HOME/git/skylar"
# atom $PROJECT/docker/.env

sed -i -e 's|ENV_PULSE_AUDIO_TCP=.*|ENV_PULSE_AUDIO_TCP=PULSE_SERVER=192.168.0.10:4713|' $PROJECT/docker/.env
cat $PROJECT/docker/.env | grep ENV_PULSE_AUDIO_TCP
```

Analyse mit R

RNeo4j - analyse

```
# https://github.com/nicolewhite/RNeo4j
library(RNeo4j)

# connect to database
graph = startGraph("http://skylar.livingfire.de/db/data/", username="skylar", password="skylar")


# define query
query <- "
MATCH (n:Listdata)
WHERE n.group = 'running' AND n.dimension = 'distance'
RETURN n
"

# retrieve nodes as List
nodes <- cypherToList(graph, query)

# retrieve vector of values
values <- sapply(nodes, function(node) node$n$value)
```

Plotten mit visNetwork - analyse

Read article: Use visNetwork to visualize your Neo4j Schema [http://www.livingfire.de/proggen/neo4j-graph-schema-visualization-with-netviz-en/]

Plotten mit Plotly - analyse

```
# https://github.com/nicolewhite/RNeo4j
library(RNeo4j)

# connect to database
graph = startGraph("http://skylar.livingfire.de/db/data/", username="skylar", password="skylar")


# define query
query <- "
MATCH (n:Listdata)
WHERE n.group = 'running'
AND n.dimension IN ['speed','distance','timeTotal']
RETURN  n
"

# retrieve nodes as List
nodes <- cypherToList(graph, query)

# setup plotData
plotData <- data.frame()
for(node in nodes) {
  plotData[node$n$logstash, node$n$dimension] <- node$n$value
}
plotData <- cbind(dateTime = rownames(plotData), plotData)

# convert to R data types
plotData$dateTime <- as.POSIXct(plotData$dateTime, "%Y-%m-%dT%H:%M:%S", tz="UTC")
plotData$distance <- as.numeric(plotData$distance)
plotData$timeTotal <- as.numeric(plotData$timeTotal)
plotData$speed <- as.numeric(plotData$speed)


# https://plot.ly/r/getting-started/
library(plotly)

axisX <- list(
  title="time"
```

```
)

axisY <- list(
  tickfont = list(color = "#4885ed"),
  title = "speed",
  titlefont = list(color = "#4885ed")
)

axisY2 <- list(
  tickfont = list(color = "#ff7f0e"),
  title = "distance",
  titlefont = list(color = "#ff7f0e"),
  position = 0.10,
  anchor = "free",
  overlaying = "y",
  side = "left"
)

axisY3 <- list(
  tickfont = list(color = "#3cba54"),
  title = "totalTime",
  titlefont = list(color = "#3cba54"),
  position = 0.20,
  anchor = "free",
  overlaying = "y",
  side = "left"
)

# create plot
p <- plot_ly(plotData, x = ~dateTime) %>%
  add_lines(y = ~speed, name = "speed", line = list(shape = "linear")) %>%
  add_lines(y = ~distance, yaxis = "y2", name = "distance", line = list(shape = "linear")) %>%
  add_lines(y = ~timeTotal, yaxis = "y3", name = "timeTotal", line = list(shape = "linear"))  %>%
  layout(
    title = "Jogging",
    yaxis = axisY,
    yaxis2 = axisY2,
    yaxis3 = axisY3,
    xaxis = axisX
  )

# show plot
p

# save plot
htmlwidgets::saveWidget(as_widget(p), "/tmp/graph.html")
```

Entwicklung

Entwicklungsumgebung skylar-client

```
PROJECT="$HOME/git/skylar"

# install nodejs and npm
apt install nodejs npm

# start nodejs server
    cd $PROJECT/project/skylar-client \
&&  npm start

# build js file
npm run build
```

Dokumentation HTML - entwicklung

Skylar hat eine mehrsprachige HTML Dokumentation in der Sprache Markdown [https://de.wikipedia.org/wiki/Markdown] die mit Jekyll [http://jekyllrb.com/] generiert wird.

```
PROJECT="$HOME/git/skylar"

# open web browser
firefox --private-window http://localhost:4000/en/manual &

# start development ->  page-prefix: /
    cd $PROJECT/docs \
&&  sed -i 's|^page-prefix:.*|page-prefix: /|' _config.yml \
&&  docker-compose up && docker-compose down

# * reload webpage when jekyll finished booting up
# * hit STRG+C to stop jekyll

# deactivate development mode ->  page-prefix: /skylar-the-scholar/
    cd $PROJECT/docs \
&&  sed -i 's|^page-prefix:.*|page-prefix: /skylar-the-scholar/|' _config.yml
```

Artikel und Jekyll config.yml

```
PROJECT="$HOME/git/skylar"
# edit documentation
atom $PROJECT/docs/_includes

# changes to _config.yml require docker restart
cd $PROJECT/docs
atom _config.yml
docker-compose restart
```

Dokumentation PDF - entwicklung

Skylar verwendet die Markdown [https://en.wikipedia.org/wiki/Markdown] Dateien der HTML Dokumentation um mit livingfire-docbook [https://github.com/phoen1x/livingfire-docbook] ein PDF in Buchqualität zu erzeugen.

```
PROJECT="$HOME/git/skylar"

# rebuild pdf after Markdown change
```

```
    cd $PROJECT/book \
&&  ./buildSklarManual.sh

# edit book
atom $PROJECT/book/book/src/main/documentation
```

Mehr Informationen hierzu finden sich auf der livingfire-docbook Projekt Seite [https://www.livingfire.de/en/docbook/#]

Dokumentation API - entwicklung

Die Skylar API wird mit Spring REST Docs [http://docs.spring.io/spring-restdocs/docs/current/reference/html5/] dokumentiert

```
PROJECT="$HOME/git/skylar"
API_PROJECT="$PROJECT/project/skylar-core"
POM_FILE="$API_PROJECT/pom.xml"

# open ascidoc configuration
atom $API_PROJECT/src/main/asciidoc

# show configuration
grep -i -B 1 -A 10 'generate-api-docs-html' $POM_FILE
grep -i -B 1 -A 10 'generate-api-docs-docbook' $POM_FILE

# change pom
eclipse $POM_FILE

# build documentation
cd $API_PROJECT
mvn clean package
```

Boostrap Themes - entwicklung

Das Webinterface verwendet Bootstrap [http://getbootstrap.com/] das durch ein config.json modifiziert werden kann.

```
PROJECT="$HOME/git/skylar"

# edit config.json
atom $PROJECT/project/skylar-client/public/assets/theme/dark/config.json
```

Um eine Boostrap Theme .zip Datei zu erhalten muss man das config.json auf die Boostrap customization Webseite [http://getbootstrap.com/customize/] hochladen.

Release - entwicklung

Erstellen der Binär Dateien

```
PROJECT="$HOME/git/skylar"
cd $PROJECT
./container_start.sh build

# test webpage
firefox --private-window "http://skylar.livingfire.de"

# stop Skylar
./container_destroy.sh
```

API Dokumentation erstellen

```
```

```
cd $PROJECT/project
./mvnw clean install
firefox --private-window "$PROJECT/docs/api/skylar-core/index.html"
```

Erstellen der HTML Dokumentation

see documentation

Erstellen der PDF Dokumentation

see documentation

API

skylar-core API

using Spring REST Docs

> **Tip**
>
> This API uses HAL [https://en.wikipedia.org/wiki/Hypertext_Application_Language] / Spring HATEOAS [http://docs.spring.io/spring-data/rest/docs/current/reference/html]. If you are unfamiliar with Hypermedia Controls [https://martinfowler.com/articles/richardsonMaturityModel.html] then I highly recommend to watch this video [https://www.youtube.com/watch?v=aThIuSsb_OA] before you explore the documentation.

Entity

Skylar uses a DAO [https://en.wikipedia.org/wiki/Data_access_object] structure which will be referred as Entity in this documentation.

```
{
    "id": 1,
    "uuid" : "00000000-eaf1-4c58-baaf-e29b38c961f6",
    "logstash" : "2017-04-01T00:00:00+02:00",
    "description" : "Thomas",
    "...": "..."
}
```

This API mainly focuses on performing CRUD operations [https://en.wikipedia.org/wiki/Create,_read,_update_and_delete] to these Entities which share the following properties

Name	Type	Usage
id	Long [https://docs.oracle.com/javase/tutorial/java/nutsandbolts/datatypes.html]	local reference id. E.g. Neo4j database id which is ONLY unique in Neo4j scope.
uuid	UUID [https://en.wikipedia.org/wiki/Universally_unique_identifier]	global reference id. Unique across the application scope. E.g. when using this identifier to search in Neo4j and Elasticsearch. You will get the Neo4j and Elasticsearch representation of the same Entity
logstash	String [https://docs.oracle.com/javase/tutorial/java/nutsandbolts/datatypes.html]	Log or creation date of the Entity. For more information see Logstash [https://www.elastic.co/products/logstash] or

Name	Type	Usage
		ElasticsearchUtil [https://github.com/phoen1x/skylar-the-scholar/blob/master/project/skylar-library/src/main/java/de/livingfire/skylar/util/ElasticsearchUtil.java].
description	String [https://docs.oracle.com/javase/tutorial/java/nutsandbolts/datatypes.html]	Description or name of a Entity

Entity types

<iframe></iframe>

Scope user session

- Setting
- User

Scope list

- ListData

Scope job application

- Contact
- Firm
- Note
- Technology

Index

Response fields

Path	Type	Description
`_links`	`Object`	Links to other resources

Example request

```
$ curl 'http://skylar.livingfire.de/api' -i -X GET
```

Example response

```
HTTP/1.1 200 OK
Content-Type: application/hal+json;charset=UTF-8
Content-Length: 1445
```

```
{
  "_links" : {
    "contacts" : {
      "href" : "http://skylar.livingfire.de/api/contacts{?page,size,sort}",
      "templated" : true
    },
    "users" : {
      "href" : "http://skylar.livingfire.de/api/users{?page,size,sort}",
      "templated" : true
    },
    "firms" : {
      "href" : "http://skylar.livingfire.de/api/firms{?page,size,sort}",
      "templated" : true
    },
    "settings" : {
      "href" : "http://skylar.livingfire.de/api/settings{?page,size,sort}",
      "templated" : true
    },
    "dimensions" : {
      "href" : "http://skylar.livingfire.de/api/dimensions{?page,size,sort}",
      "templated" : true
    },
    "technologys" : {
      "href" : "http://skylar.livingfire.de/api/technologys{?page,size,sort}",
      "templated" : true
    },
    "notes" : {
      "href" : "http://skylar.livingfire.de/api/notes{?page,size,sort}",
      "templated" : true
    },
    "listdatas" : {
      "href" : "http://skylar.livingfire.de/api/listdatas{?page,size,sort}",
      "templated" : true
    },
    "javascript" : {
      "href" : "http://skylar.livingfire.de/api/javascript"
    },
    "export" : {
      "href" : "http://skylar.livingfire.de/api/export"
    },
    "import" : {
      "href" : "http://skylar.livingfire.de/api/import"
    },
    "selenium" : {
      "href" : "http://skylar.livingfire.de/api/selenium"
    },
    "profile" : {
      "href" : "http://skylar.livingfire.de/api/profile"
    }
  }
}
```


Links

Relation	Description
`users`	See Users
`listdatas`	See ListDatas
`contacts`	See Contacts
`firms`	See Firms
`notes`	See Notes
`technologys`	See Technologies
`dimensions`	See Dimensions
`settings`	See Settings
`javascript`	Helper methods Javascript
`export`	Helper methods Export
`import`	Helper methods Import

Relation	Description
`selenium`	See Selenium
`profile`	The Application-Level Profile Semantics (ALPS)

Error

Response fields

Path	Type	Description
`error`	`String`	The HTTP status code that occurred
`message`	`String`	A massage like *`Bad Request`*
`path`	`String`	The path to which the request was made
`status`	`Number`	The HTTP status code like *`400`*
`timestamp`	`String`	The time in milliseconds at which the error occurred

Example request

```
$ curl 'http://skylar.livingfire.de/error' -i -X GET
```

Example response

```
HTTP/1.1 400 Bad Request
Content-Type: application/json;charset=UTF-8
Content-Length: 157

{
  "timestamp" : "2019-01-28T15:08:07.836+0000",
  "status" : 400,
  "error" : "Bad Request",
  "message" : "Bad Request",
  "path" : "/repository/fooBar"
}
```

Contact

Details

See Application-Level Profile Semantics (ALPS) [http://docs.spring.io/spring-data/rest/docs/current/reference/html/#metadata.alps]

Example request

```
$ curl 'http://skylar.livingfire.de/api/profile/contacts' -i -X GET
```

Create

A `POST` request will create a Contact.

Example request

```
$ curl 'http://skylar.livingfire.de/api/contacts' -i -X POST \
    -H 'Content-Type: application/hal+json' \
    -d '{
  "id" : null,
  "uuid" : "8e5c0df2-0ba6-4c74-8393-d38ec198e754",
  "logstash" : "2017-04-01T00:00:00+02:00",
  "description" : "Jon Doe",
  "email" : "gibts@gar.net",
  "phone" : "555-123456",
  "recruiter" : false
}'
```

Example response

```
HTTP/1.1 201 Created
Location: http://skylar.livingfire.de/api/contacts/0
```

Retrieve

A `GET` request will retrieve a Contact.

Response fields

Path	Type	Description
`uuid`	`String`	Java UUID
`logstash`	`String`	ISO 8601 date with TimeZone UTC in format "yyyy-MM-dd'T'HH:mm:ssZ" e.g. 2017-04-01T00:00:00Z
`description`	`String`	Description or name
`email`	`String`	a email address like 'gibts@gar.net'
`phone`	`String`	a phone number like '555-123456'
`recruiter`	`Boolean`	true == external recruiter firm
`_links`	`Object`	Links to other resources

Example request

```
$ curl 'http://skylar.livingfire.de/api/contacts/0' -i -X GET
```

Example response

```
HTTP/1.1 200 OK
Content-Type: application/hal+json;charset=UTF-8
Content-Length: 390

{
  "uuid" : "8e5c0df2-0ba6-4c74-8393-d38ec198e754",
  "logstash" : "2017-04-01T00:00:00+02:00",
  "description" : "Jon Doe",
```

```
  "email" : "gibts@gar.net",
  "phone" : "555-123456",
  "recruiter" : false,
  "_links" : {
    "self" : {
      "href" : "http://skylar.livingfire.de/api/contacts/0"
    },
    "contact" : {
      "href" : "http://skylar.livingfire.de/api/contacts/0"
    }
  }
}
```

Links

Relation	Description
`self`	Canonical link for this resource
`contact`	This Contact

Update

A `PATCH` request will update a Contact.

Example request

```
$ curl 'http://skylar.livingfire.de/api/contacts/0' -i -X PATCH \
    -H 'Content-Type: application/hal+json' \
    -d '{
  "description" : "John Doe"
}'
```

Example response

```
HTTP/1.1 204 No Content
```

Delete

A `DELETE` request will delete a Contact.

Example request

```
$ curl 'http://skylar.livingfire.de/api/contacts/0' -i -X DELETE \
    -H 'Content-Type: application/hal+json'
```

Example response

```
HTTP/1.1 204 No Content
```

Search

Details

See Application-Level Profile Semantics (ALPS) [http://docs.spring.io/spring-data/rest/docs/current/reference/html/#metadata.alps]

Example request

```
$ curl 'http://skylar.livingfire.de/api/contacts/search' -i -X GET
```

Example response

```
HTTP/1.1 200 OK
```

```
Content-Type: application/hal+json;charset=UTF-8
Content-Length: 403

{
  "_links" : {
    "findByDescription" : {
      "href" : "http://skylar.livingfire.de/api/contacts/search/findByDescription{?description}",
      "templated" : true
    },
    "findByUuid" : {
      "href" : "http://skylar.livingfire.de/api/contacts/search/findByUuid{?uuid}",
      "templated" : true
    },
    "self" : {
      "href" : "http://skylar.livingfire.de/api/contacts/search"
    }
  }
}
```

findByUuid

A `GET` request will retrieve a Contact.

Request parameters

Parameter	Description
`uuid`	Java UUID

Response fields

Path	Type	Description
`uuid`	`String`	Java UUID
`logstash`	`String`	ISO 8601 date with TimeZone UTC in format "yyyy-MM-dd'T'HH:mm:ssZ" e.g. 2017-04-01T00:00:00Z
`description`	`String`	Description or name
`email`	`String`	a email address like 'gibts@gar.net'
`phone`	`String`	a phone number like '555-123456'
`recruiter`	`Boolean`	true == external recruiter firm
`_links`	`Object`	Links to other resources

Example request

```
$ curl 'http://skylar.livingfire.de/api/contacts/search/findByUuid?uuid=8e5c0df2-0ba6-4c74-8393-d38ec198e754' -i -X GET
```

Example response

```
HTTP/1.1 200 OK
Content-Type: application/hal+json;charset=UTF-8
Content-Length: 391

{
  "uuid" : "8e5c0df2-0ba6-4c74-8393-d38ec198e754",
  "logstash" : "2017-04-01T00:00:00+02:00",
  "description" : "John Doe",
```

```
  "email" : "gibts@gar.net",
  "phone" : "555-123456",
  "recruiter" : false,
  "_links" : {
    "self" : {
      "href" : "http://skylar.livingfire.de/api/contacts/0"
    },
    "contact" : {
      "href" : "http://skylar.livingfire.de/api/contacts/0"
    }
  }
}
```

Links

Relation	Description
`self`	Canonical link for this resource
`contact`	This Contact

findByDescription

A `GET` request will retrieve a array of Contact.

Request parameters

Parameter	Description
`description`	Description or name

Response fields

Path	Type	Description
`_embedded.contacts`	`Array`	An array of Contact
`_links`	`Object`	Links to other resources

Example request

```
$ curl 'http://skylar.livingfire.de/api/contacts/search/findByDescription?description=Jon+Doe' -i -X GET
```

Example response

```
HTTP/1.1 200 OK
Content-Type: application/hal+json;charset=UTF-8
Content-Length: 170

{
  "_embedded" : {
    "contacts" : [ ]
  },
  "_links" : {
    "self" : {
      "href" : "http://skylar.livingfire.de/api/contacts/search/findByDescription"
    }
  }
}
```

Links

Relation	Description
`self`	Canonical link for this resource

Contacts

Listing

A `GET` request will retrieve a paginated view [http://docs.spring.io/spring-data/rest/docs/current/reference/html/#paging-and-sorting] of all Contact.

Response fields

Path	Type	Description
`_embedded.contacts`	`Array`	An array of Contact
`_links`	`Object`	Links to other resources
`page`	`Object`	paging information

Example request

```
$ curl 'http://skylar.livingfire.de/api/contacts' -i -X GET
```

Example response

```
HTTP/1.1 200 OK
Content-Type: application/hal+json;charset=UTF-8
Content-Length: 919

{
  "_embedded" : {
    "contacts" : [ {
      "uuid" : "8e5c0df2-0ba6-4c74-8393-d38ec198e754",
      "logstash" : "2017-04-01T00:00:00+02:00",
      "description" : "John Doe",
      "email" : "gibts@gar.net",
      "phone" : "555-123456",
      "recruiter" : false,
      "_links" : {
        "self" : {
          "href" : "http://skylar.livingfire.de/api/contacts/0"
        },
        "contact" : {
          "href" : "http://skylar.livingfire.de/api/contacts/0"
        }
      }
    } ]
  },
  "_links" : {
    "self" : {
      "href" : "http://skylar.livingfire.de/api/contacts{?page,size,sort}",
      "templated" : true
    },
    "profile" : {
      "href" : "http://skylar.livingfire.de/api/profile/contacts"
    },
    "search" : {
      "href" : "http://skylar.livingfire.de/api/contacts/search"
    }
  },
  "page" : {
    "size" : 20,
    "totalElements" : 1,
    "totalPages" : 1,
    "number" : 0
  }
}
```

Links

Relation	Description
`self`	Canonical link for this resource

Relation	Description
`profile`	The Application-Level Profile Semantics (ALPS)
`search`	Canonical link to search resources

Firm

Details

See Application-Level Profile Semantics (ALPS) [http://docs.spring.io/spring-data/rest/docs/current/reference/html/#metadata.alps]

Example request

```
$ curl 'http://skylar.livingfire.de/api/profile/firms' -i -X GET
```

Create

A `POST` request will create a Firm.

Example request

```
$ curl 'http://skylar.livingfire.de/api/firms' -i -X POST \
    -H 'Content-Type: application/hal+json' \
    -d '{
  "id" : null,
  "uuid" : "2d845b50-74cb-4d18-b1bb-c7b969024f66",
  "logstash" : "2017-04-01T00:00:00+02:00",
  "description" : "Google",
  "active" : true,
  "url" : "http://www.google.de",
  "postalCode" : "12345",
  "city" : "ACME city",
  "street" : "ACME street 5",
  "relationRecruiter" : {
    "id" : null,
    "uuid" : "7d06d925-4e61-4648-9679-49b95c88fdec",
    "logstash" : "2017-04-01T00:00:00+02:00",
    "description" : "John Doe",
    "email" : "gibts@gar.net",
    "phone" : "555-123456",
    "recruiter" : false
  },
  "relationTechnologies" : [ {
    "id" : null,
    "uuid" : "fde1b0dc-e49c-4073-afc9-1ac215104661",
    "logstash" : "2017-04-01T00:00:00+02:00",
    "description" : "Java"
  } ],
  "relationNotes" : [ {
    "id" : null,
    "uuid" : "26d78edc-b8d2-485b-b1bf-f2f9c0651256",
    "logstash" : "2017-04-01T00:00:00+02:00",
    "description" : "2017-04-01 cycle started - Java Webdeveloper",
    "url" : "http://www.google.de/webdeveloper"
  } ]
}'
```

Example response

```
HTTP/1.1 201 Created
Location: http://skylar.livingfire.de/api/firms/1
```

Retrieve

A `GET` request will retrieve a Firm.

Response fields

Path	Type	Description
`uuid`	`String`	Java UUID
`logstash`	`String`	ISO 8601 date with TimeZone UTC in format "yyyy-MM-dd'T'HH:mm:ssZ" e.g. 2017-04-01T00:00:00Z
`description`	`String`	Description or name
`active`	`Boolean`	Active state e.g. 'true'
`url`	`String`	a URL like https://www.youtube.com/
`postalCode`	`String`	a postal code like 12345
`city`	`String`	a city name like 'ACME city'
`street`	`String`	a street address like 'ACME street 5'
`_links`	`Object`	Links to other resources

Example request

```
$ curl 'http://skylar.livingfire.de/api/firms/1' -i -X GET
```

Example response

```
HTTP/1.1 200 OK
Content-Type: application/hal+json;charset=UTF-8
Content-Length: 765

{
  "uuid" : "7d06d925-4e61-4648-9679-49b95c88fdec",
  "logstash" : "2017-04-01T00:00:00+02:00",
  "description" : "John Doe",
  "active" : true,
  "url" : "http://www.google.de",
  "postalCode" : "12345",
  "city" : "ACME city",
  "street" : "ACME street 5",
  "_links" : {
    "self" : {
      "href" : "http://skylar.livingfire.de/api/firms/1"
    },
    "firm" : {
      "href" : "http://skylar.livingfire.de/api/firms/1"
    },
    "relationTechnologies" : {
      "href" : "http://skylar.livingfire.de/api/firms/1/relationTechnologies"
    },
    "relationRecruiter" : {
      "href" : "http://skylar.livingfire.de/api/firms/1/relationRecruiter"
    },
    "relationNotes" : {
      "href" : "http://skylar.livingfire.de/api/firms/1/relationNotes"
    }
  }
}
```

Links

Relation	Description
`self`	Canonical link for this resource

Relation	Description
`firm`	This Firm
`relationNotes`	Relation Notes
`relationTechnologies`	Relation Technologies
`relationRecruiter`	Relation Contact

Update

A `PATCH` request will update a Firm.

Example request

```
$ curl 'http://skylar.livingfire.de/api/firms/1' -i -X PATCH \
    -H 'Content-Type: application/hal+json' \
    -d '{
  "description" : "Google"
}'
```

Example response

```
HTTP/1.1 204 No Content
```

Delete

A `DELETE` request will delete a Firm.

Example request

```
$ curl 'http://skylar.livingfire.de/api/firms/1' -i -X DELETE \
    -H 'Content-Type: application/hal+json'
```

Example response

```
HTTP/1.1 204 No Content
```

Search

Details

See Application-Level Profile Semantics (ALPS) [http://docs.spring.io/spring-data/rest/docs/current/reference/html/#metadata.alps]

Example request

```
$ curl 'http://skylar.livingfire.de/api/firms/search' -i -X GET
```

Example response

```
HTTP/1.1 200 OK
Content-Type: application/hal+json;charset=UTF-8
Content-Length: 534

{
  "_links" : {
    "findByDescription" : {
      "href" : "http://skylar.livingfire.de/api/firms/search/findByDescription{?description}",
      "templated" : true
    },
    "findByActive" : {
      "href" : "http://skylar.livingfire.de/api/firms/search/findByActive{?active}",
      "templated" : true
```

```
    },
    "findByUuid" : {
      "href" : "http://skylar.livingfire.de/api/firms/search/findByUuid{?uuid}",
      "templated" : true
    },
    "self" : {
      "href" : "http://skylar.livingfire.de/api/firms/search"
    }
  }
}
```

findByUuid

A `GET` request will retrieve a Firm.

Request parameters

Parameter	Description
`uuid`	Java UUID

Response fields

Path	Type	Description
`uuid`	`String`	Java UUID
`logstash`	`String`	ISO 8601 date with TimeZone UTC in format "yyyy-MM-dd'T'HH:mm:ssZ" e.g. 2017-04-01T00:00:00Z
`description`	`String`	Description or name
`active`	`Boolean`	Active state e.g. 'true'
`url`	`String`	a URL like https://www.youtube.com/
`postalCode`	`String`	a postal code like 12345
`city`	`String`	a city name like 'ACME city'
`street`	`String`	a street address like 'ACME street 5'
`_links`	`Object`	Links to other resources

Example request

```
$ curl 'http://skylar.livingfire.de/api/firms/search/findByUuid?uuid=7d06d925-4e61-4648-9679-49b95c88fdec' -i -X GET
```

Example response

```
HTTP/1.1 200 OK
Content-Type: application/hal+json;charset=UTF-8
Content-Length: 763

{
  "uuid" : "7d06d925-4e61-4648-9679-49b95c88fdec",
  "logstash" : "2017-04-01T00:00:00+02:00",
  "description" : "Google",
  "active" : true,
  "url" : "http://www.google.de",
  "postalCode" : "12345",
```

```
  "city" : "ACME city",
  "street" : "ACME street 5",
  "_links" : {
    "self" : {
      "href" : "http://skylar.livingfire.de/api/firms/1"
    },
    "firm" : {
      "href" : "http://skylar.livingfire.de/api/firms/1"
    },
    "relationTechnologies" : {
      "href" : "http://skylar.livingfire.de/api/firms/1/relationTechnologies"
    },
    "relationRecruiter" : {
      "href" : "http://skylar.livingfire.de/api/firms/1/relationRecruiter"
    },
    "relationNotes" : {
      "href" : "http://skylar.livingfire.de/api/firms/1/relationNotes"
    }
  }
}
```

Links

Relation	Description
`self`	Canonical link for this resource
`firm`	This Firm
`relationNotes`	Relation Notes
`relationTechnologies`	Relation Technologies
`relationRecruiter`	Relation Contact

findByDescription

A `GET` request will retrieve a array of Firm.

Request parameters

Parameter	Description
`description`	Description or name

Response fields

Path	Type	Description
`_embedded.firms`	`Array`	An array of Firm
`_links`	`Object`	Links to other resources

Example request

```
$ curl 'http://skylar.livingfire.de/api/firms/search/findByDescription?description=John+Doe' -i -X GET
```

Example response

```
HTTP/1.1 200 OK
Content-Type: application/hal+json;charset=UTF-8
Content-Length: 164

{
  "_embedded" : {
    "firms" : [ ]
  },
  "_links" : {
    "self" : {
```

```
      "href" : "http://skylar.livingfire.de/api/firms/search/findByDescription"
    }
  }
}
```

Links

Relation	Description
`self`	Canonical link for this resource

Firms

Listing

A `GET` request will retrieve a paginated view [http://docs.spring.io/spring-data/rest/docs/current/reference/html/#paging-and-sorting] of all Firm.

Response fields

Path	Type	Description
`_embedded.firms`	`Array`	An array of Firm
`_links`	`Object`	Firms to other resources
`page`	`Object`	paging information

Example request

```
$ curl 'http://skylar.livingfire.de/api/firms' -i -X GET
```

Example response

```
HTTP/1.1 200 OK
Content-Type: application/hal+json;charset=UTF-8
Content-Length: 1323

{
  "_embedded" : {
    "firms" : [ {
      "uuid" : "7d06d925-4e61-4648-9679-49b95c88fdec",
      "logstash" : "2017-04-01T00:00:00+02:00",
      "description" : "Google",
      "active" : true,
      "url" : "http://www.google.de",
      "postalCode" : "12345",
      "city" : "ACME city",
      "street" : "ACME street 5",
      "_links" : {
        "self" : {
          "href" : "http://skylar.livingfire.de/api/firms/1"
        },
        "firm" : {
          "href" : "http://skylar.livingfire.de/api/firms/1"
        },
        "relationTechnologies" : {
          "href" : "http://skylar.livingfire.de/api/firms/1/relationTechnologies"
        },
        "relationRecruiter" : {
          "href" : "http://skylar.livingfire.de/api/firms/1/relationRecruiter"
        },
        "relationNotes" : {
          "href" : "http://skylar.livingfire.de/api/firms/1/relationNotes"
        }
      }
    } ]
  },
```

```
  "_links" : {
    "self" : {
      "href" : "http://skylar.livingfire.de/api/firms{?page,size,sort}",
      "templated" : true
    },
    "profile" : {
      "href" : "http://skylar.livingfire.de/api/profile/firms"
    },
    "search" : {
      "href" : "http://skylar.livingfire.de/api/firms/search"
    }
  },
  "page" : {
    "size" : 20,
    "totalElements" : 1,
    "totalPages" : 1,
    "number" : 0
  }
}
```

Links

Relation	Description
`self`	Canonical link for this resource
`profile`	The Application-Level Profile Semantics (ALPS)
`search`	Canonical link to search resources

ListData

Details

See Application-Level Profile Semantics (ALPS) [http://docs.spring.io/spring-data/rest/docs/current/reference/html/#metadata.alps]

Example request

```
$ curl 'http://skylar.livingfire.de/api/profile/listdatas' -i -X GET
```

Create

A `POST` request will create a ListData.

Example request

```
$ curl 'http://skylar.livingfire.de/api/listdatas' -i -X POST \
    -H 'Content-Type: application/hal+json' \
    -d '{
  "id" : null,
  "uuid" : "dc06c6d1-39d9-4869-8628-938dd0fa0375",
  "logstash" : "2017-04-01T00:00:01Z",
  "description" : "beautiful weather and good run",
  "group" : "running",
  "entryDate" : "2019-01-28T15:08:11.752Z",
  "dimension" : "distance",
  "unit" : "km",
  "value" : "13",
  "relationDimensionHasDimension" : null
}'
```

Example response

```
HTTP/1.1 201 Created
Location: http://skylar.livingfire.de/api/listdatas/2
```

Retrieve

A `GET` request will retrieve a ListData.

Response fields

Path	Type	Description
`uuid`	`String`	Java UUID
`logstash`	`String`	ISO 8601 date with TimeZone UTC in format "yyyy-MM-dd'T'HH:mm:ssZ" e.g. 2017-04-01T00:00:00Z
`description`	`String`	Description or name
`dimension`	`String`	Name of a dimension in a matrix. E.g. distance, time, x, y, z, ...
`group`	`String`	The group this value belongs to. This can be thougt of a as a another dimension in a matrix
`entryDate`	`String`	creation date
`unit`	`String`	a unit of measurement like km, s, ccm
`value`	`String`	Value like 42, 3.1415, 'Hello World!'
`_links`	`Object`	Links to other resources

Example request

```
$ curl 'http://skylar.livingfire.de/api/listdatas/2' -i -X GET
```

Example response

```
HTTP/1.1 200 OK
Content-Type: application/hal+json;charset=UTF-8
Content-Length: 601

{
  "uuid" : "dc06c6d1-39d9-4869-8628-938dd0fa0375",
  "logstash" : "2017-04-01T00:00:01Z",
  "description" : "beautiful weather and good run",
  "group" : "running",
  "entryDate" : "2019-01-28T15:08:11.752Z",
  "dimension" : "distance",
  "unit" : "km",
  "value" : "13",
  "_links" : {
    "self" : {
      "href" : "http://skylar.livingfire.de/api/listdatas/2"
    },
    "listdata" : {
      "href" : "http://skylar.livingfire.de/api/listdatas/2"
    },
    "relationDimensionHasDimension" : {
      "href" : "http://skylar.livingfire.de/api/listdatas/2/relationDimensionHasDimension"
    }
  }
}
```

Links

Relation	Description
`self`	Canonical link for this resource
`listdata`	This ListData
`relationDimensionHasDimension`	Relation Dimension

Update

A `PATCH` request will update a ListData.

Example request

```
$ curl 'http://skylar.livingfire.de/api/listdatas/2' -i -X PATCH \
    -H 'Content-Type: application/hal+json' \
    -d '{
  "description" : "bad rain on run"
}'
```

Example response

```
HTTP/1.1 204 No Content
```

Delete

A `DELETE` request will delete a ListData.

Example request

```
$ curl 'http://skylar.livingfire.de/api/listdatas/2' -i -X DELETE \
    -H 'Content-Type: application/hal+json'
```

Example response

```
HTTP/1.1 204 No Content
```

Search

Details

See Application-Level Profile Semantics (ALPS) [http://docs.spring.io/spring-data/rest/docs/current/reference/html/#metadata.alps]

Example request

```
$ curl 'http://skylar.livingfire.de/api/listdatas/search' -i -X GET
```

Example response

```
HTTP/1.1 200 OK
Content-Type: application/hal+json;charset=UTF-8
Content-Length: 835

{
  "_links" : {
    "findByUuid" : {
      "href" : "http://skylar.livingfire.de/api/listdatas/search/findByUuid{?uuid}",
      "templated" : true
    },
    "findByDescription" : {
      "href" : "http://skylar.livingfire.de/api/listdatas/search/findByDescription{?description}",
      "templated" : true
    },
    "findByGroupAndDimension" : {
```

```
      "href" : "http://skylar.livingfire.de/api/listdatas/search/findByGroupAndDimension{?group,dimension}",
      "templated" : true
    },
    "groupDimension" : {
      "href" : "http://skylar.livingfire.de/api/listdatas/search/groupDimension"
    },
    "findByGroup" : {
      "href" : "http://skylar.livingfire.de/api/listdatas/search/findByGroup{?group}",
      "templated" : true
    },
    "self" : {
      "href" : "http://skylar.livingfire.de/api/listdatas/search"
    }
  }
}
```

findByUuid

A `GET` request will retrieve a ListData.

Request parameters

Parameter	Description
`uuid`	Java UUID

Response fields

Path	Type	Description
`uuid`	`String`	Java UUID
`logstash`	`String`	ISO 8601 date with TimeZone UTC in format "yyyy-MM-dd'T'HH:mm:ssZ" e.g. 2017-04-01T00:00:00Z
`description`	`String`	Description or name
`dimension`	`String`	Name of a dimension in a matrix. E.g. distance, time, x, y, z, ...
`group`	`String`	The group this value belongs to. This can be thougt of a as a another dimension in a matrix
`entryDate`	`String`	creation date
`unit`	`String`	a unit of measurement like km, s, ccm
`value`	`String`	Value like 42, 3.1415, 'Hello World!'
`_links`	`Object`	Links to other resources

Example request

```
$ curl 'http://skylar.livingfire.de/api/listdatas/search/findByUuid?uuid=dc06c6d1-39d9-4869-8628-938dd0fa0375' -i -X GET
```

Example response

```
HTTP/1.1 200 OK
Content-Type: application/hal+json;charset=UTF-8
```

```
Content-Length: 586

{
  "uuid" : "dc06c6d1-39d9-4869-8628-938dd0fa0375",
  "logstash" : "2017-04-01T00:00:01Z",
  "description" : "bad rain on run",
  "group" : "running",
  "entryDate" : "2019-01-28T15:08:11.752Z",
  "dimension" : "distance",
  "unit" : "km",
  "value" : "13",
  "_links" : {
    "self" : {
      "href" : "http://skylar.livingfire.de/api/listdatas/2"
    },
    "listdata" : {
      "href" : "http://skylar.livingfire.de/api/listdatas/2"
    },
    "relationDimensionHasDimension" : {
      "href" : "http://skylar.livingfire.de/api/listdatas/2/relationDimensionHasDimension"
    }
  }
}
```

Links

Relation	Description
`self`	Canonical link for this resource
`listdata`	This ListData
`relationDimensionHasDimension`	Relation Dimension

findByDescription

A `GET` request will retrieve a array of ListData.

Request parameters

Parameter	Description
`description`	Description or name

Response fields

Path	Type	Description
`_embedded.listdatas`	`Array`	An array of ListData
`_links`	`Object`	Links to other resources

Example request

```
$ curl 'http://skylar.livingfire.de/api/listdatas/search/findByDescription?description=beautiful+weather+and+good+run' -i -X GET
```

Example response

```
HTTP/1.1 200 OK
Content-Type: application/hal+json;charset=UTF-8
Content-Length: 172

{
  "_embedded" : {
    "listdatas" : [ ]
  },
  "_links" : {
    "self" : {
      "href" : "http://skylar.livingfire.de/api/listdatas/search/findByDescription"
    }
  }
}
```

Links

Relation	Description
`self`	Canonical link for this resource

ListDatas

Listing

A `GET` request will retrieve a paginated view [http://docs.spring.io/spring-data/rest/docs/current/reference/html/#paging-and-sorting] of all Setting.

Response fields

Path	Type	Description
`_embedded.settings`	`Array`	An array of Setting
`_links`	`Object`	Links to other resources
`page`	`Object`	paging information

Example request

```
$ curl 'http://skylar.livingfire.de/api/settings' -i -X GET
```

Example response

```
HTTP/1.1 200 OK
Content-Type: application/hal+json;charset=UTF-8
Content-Length: 1007

{
  "_embedded" : {
    "settings" : [ {
      "uuid" : "85f4fb14-8493-4ceb-945d-cdbd4f239cb9",
      "logstash" : "2017-04-01T00:00:00+02:00",
      "description" : "en",
      "dimension" : "ttsLanguage",
      "_links" : {
        "self" : {
          "href" : "http://skylar.livingfire.de/api/settings/4"
        },
        "setting" : {
          "href" : "http://skylar.livingfire.de/api/settings/4"
        },
        "relationDimensionHasDimension" : {
          "href" : "http://skylar.livingfire.de/api/settings/4/relationDimensionHasDimension"
        }
      }
    } ]
  },
  "_links" : {
    "self" : {
      "href" : "http://skylar.livingfire.de/api/settings{?page,size,sort}",
      "templated" : true
    },
    "profile" : {
      "href" : "http://skylar.livingfire.de/api/profile/settings"
    },
    "search" : {
      "href" : "http://skylar.livingfire.de/api/settings/search"
    }
  },
  "page" : {
    "size" : 20,
    "totalElements" : 1,
    "totalPages" : 1,
    "number" : 0
  }
```

```
}
```

Links

Relation	Description
`self`	Canonical link for this resource
`profile`	The Application-Level Profile Semantics (ALPS)
`search`	Canonical link to search resources

Note

Details

See Application-Level Profile Semantics (ALPS) [http://docs.spring.io/spring-data/rest/docs/current/reference/html/#metadata.alps]

Example request

```
$ curl 'http://skylar.livingfire.de/api/profile/notes' -i -X GET
```

Create

A `POST` request will create a Note.

Example request

```
$ curl 'http://skylar.livingfire.de/api/notes' -i -X POST \
    -H 'Content-Type: application/hal+json' \
    -d '{
  "id" : null,
  "uuid" : "2d8b978f-e659-43c5-96b5-563ab7164fd5",
  "logstash" : "2017-04-01T00:00:00+02:00",
  "description" : "2018-04-01 started job - Java Application Developer",
  "url" : "http://www.google.de/javaDeveloper"
}'
```

Example response

```
HTTP/1.1 201 Created
Location: http://skylar.livingfire.de/api/notes/3
```

Retrieve

A `GET` request will retrieve a Note.

Response fields

Path	Type	Description
`uuid`	`String`	Java UUID
`logstash`	`String`	ISO 8601 date with TimeZone UTC in format "yyyy-MM-dd'T'HH:mm:ssZ" e.g. 2017-04-01T00:00:00Z

Path	Type	Description
`description`	`String`	Description or name
`url`	`String`	a URL like https://www.youtube.com/
`_links`	`Object`	Links to other resources

Example request

```
$ curl 'http://skylar.livingfire.de/api/notes/3' -i -X GET
```

Example response

```
HTTP/1.1 200 OK
Content-Type: application/hal+json;charset=UTF-8
Content-Length: 395

{
  "uuid" : "2d8b978f-e659-43c5-96b5-563ab7164fd5",
  "logstash" : "2017-04-01T00:00:00+02:00",
  "description" : "2018-04-01 started job - Java Application Developer",
  "url" : "http://www.google.de/javaDeveloper",
  "_links" : {
    "self" : {
      "href" : "http://skylar.livingfire.de/api/notes/3"
    },
    "note" : {
      "href" : "http://skylar.livingfire.de/api/notes/3"
    }
  }
}
```

Links

Relation	Description
`self`	Canonical link for this resource
`note`	This Note

Update

A `PATCH` request will update a Note.

Example request

```
$ curl 'http://skylar.livingfire.de/api/notes/3' -i -X PATCH \
    -H 'Content-Type: application/hal+json' \
    -d '{
  "description" : "2018-04-01 started job - Java Application Developer"
}'
```

Example response

```
HTTP/1.1 204 No Content
```

Delete

A `DELETE` request will delete a Note.

Example request

```
$ curl 'http://skylar.livingfire.de/api/notes/3' -i -X DELETE \
    -H 'Content-Type: application/hal+json'
```

Example response

```
HTTP/1.1 204 No Content
```

Search

Details

See Application-Level Profile Semantics (ALPS) [http://docs.spring.io/spring-data/rest/docs/current/reference/html/#metadata.alps]

Example request

```
$ curl 'http://skylar.livingfire.de/api/notes/search' -i -X GET
```

Example response

```
HTTP/1.1 200 OK
Content-Type: application/hal+json;charset=UTF-8
Content-Length: 511

{
  "_links" : {
    "findByUuid" : {
      "href" : "http://skylar.livingfire.de/api/notes/search/findByUuid{?uuid}",
      "templated" : true
    },
    "findByDescription" : {
      "href" : "http://skylar.livingfire.de/api/notes/search/findByDescription{?description}",
      "templated" : true
    },
    "notesOfActiveFirms" : {
      "href" : "http://skylar.livingfire.de/api/notes/search/notesOfActiveFirms"
    },
    "self" : {
      "href" : "http://skylar.livingfire.de/api/notes/search"
    }
  }
}
```

findByUuid

A `GET` request will retrieve a Note.

Request parameters

Parameter	Description
`uuid`	Java UUID

Response fields

Path	Type	Description
`uuid`	`String`	Java UUID
`logstash`	`String`	ISO 8601 date with TimeZone UTC in format "yyyy-MM-dd'T'HH:mm:ssZ" e.g. 2017-04-01T00:00:00Z
`description`	`String`	Description or name
`url`	`String`	a URL like https://www.youtube.com/
`_links`	`Object`	Links to other resources

Example request

```
$ curl 'http://skylar.livingfire.de/api/notes/search/findByUuid?uuid=2d8b978f-e659-43c5-96b5-563ab7164fd5' -i -X GET
```

Example response

```
HTTP/1.1 200 OK
Content-Type: application/hal+json;charset=UTF-8
Content-Length: 395

{
  "uuid" : "2d8b978f-e659-43c5-96b5-563ab7164fd5",
  "logstash" : "2017-04-01T00:00:00+02:00",
  "description" : "2018-04-01 started job - Java Application Developer",
  "url" : "http://www.google.de/javaDeveloper",
  "_links" : {
    "self" : {
      "href" : "http://skylar.livingfire.de/api/notes/3"
    },
    "note" : {
      "href" : "http://skylar.livingfire.de/api/notes/3"
    }
  }
}
```

Links

Relation	Description
`self`	Canonical link for this resource
`note`	This Note

findByDescription

A `GET` request will retrieve a array of Note.

Request parameters

Parameter	Description
`description`	Description or name

Response fields

Path	Type	Description
`_embedded.notes`	`Array`	An array of Note
`_links`	`Object`	Links to other resources

Example request

```
$ curl 'http://skylar.livingfire.de/api/notes/search/findByDescription?description=2018-04-01+started+job+-+Java+Application+Developer
```

Example response

```
HTTP/1.1 200 OK
Content-Type: application/hal+json;charset=UTF-8
Content-Length: 612

{
  "_embedded" : {
    "notes" : [ {
      "uuid" : "2d8b978f-e659-43c5-96b5-563ab7164fd5",
      "logstash" : "2017-04-01T00:00:00+02:00",
      "description" : "2018-04-01 started job - Java Application Developer",
      "url" : "http://www.google.de/javaDeveloper",
      "_links" : {
        "self" : {
          "href" : "http://skylar.livingfire.de/api/notes/3"
        },
```

```
        "note" : {
          "href" : "http://skylar.livingfire.de/api/notes/3"
        }
      }
    } ]
  },
  "_links" : {
    "self" : {
      "href" : "http://skylar.livingfire.de/api/notes/search/findByDescription"
    }
  }
}
```

Links

Relation	Description
`self`	Canonical link for this resource

Notes

Listing

A `GET` request will retrieve a paginated view [http://docs.spring.io/spring-data/rest/docs/current/reference/html/#paging-and-sorting] of all Note.

Response fields

Path	Type	Description
`_embedded.notes`	`Array`	An array of Note
`_links`	`Object`	Notes to other resources
`page`	`Object`	paging information

Example request

```
$ curl 'http://skylar.livingfire.de/api/notes' -i -X GET
```

Example response

```
HTTP/1.1 200 OK
Content-Type: application/hal+json;charset=UTF-8
Content-Length: 903

{
  "_embedded" : {
    "notes" : [ {
      "uuid" : "2d8b978f-e659-43c5-96b5-563ab7164fd5",
      "logstash" : "2017-04-01T00:00:00+02:00",
      "description" : "2018-04-01 started job - Java Application Developer",
      "url" : "http://www.google.de/javaDeveloper",
      "_links" : {
        "self" : {
          "href" : "http://skylar.livingfire.de/api/notes/3"
        },
        "note" : {
          "href" : "http://skylar.livingfire.de/api/notes/3"
        }
      }
    } ]
  },
  "_links" : {
    "self" : {
      "href" : "http://skylar.livingfire.de/api/notes{?page,size,sort}",
      "templated" : true
    },
    "profile" : {
      "href" : "http://skylar.livingfire.de/api/profile/notes"
    },
    "search" : {
```

```
      "href" : "http://skylar.livingfire.de/api/notes/search"
    }
  },
  "page" : {
    "size" : 20,
    "totalElements" : 1,
    "totalPages" : 1,
    "number" : 0
  }
}
```

Links

Relation	Description
`self`	Canonical link for this resource
`profile`	The Application-Level Profile Semantics (ALPS)
`search`	Canonical link to search resources

Setting

Details

See Application-Level Profile Semantics (ALPS) [http://docs.spring.io/spring-data/rest/docs/current/reference/html/#metadata.alps]

Example request

```
$ curl 'http://skylar.livingfire.de/api/profile/settings' -i -X GET
```

Create

A `POST` request will create a Setting.

Example request

```
$ curl 'http://skylar.livingfire.de/api/settings' -i -X POST \
    -H 'Content-Type: application/hal+json' \
    -d '{
  "id" : null,
  "uuid" : "85f4fb14-8493-4ceb-945d-cdbd4f239cb9",
  "logstash" : "2017-04-01T00:00:00+02:00",
  "description" : "de",
  "dimension" : "ttsLanguage",
  "relationDimensionHasDimension" : null
}'
```

Example response

```
HTTP/1.1 201 Created
Location: http://skylar.livingfire.de/api/settings/4
```

Retrieve

A `GET` request will retrieve a Setting.

Response fields

Path	Type	Description
`uuid`	`String`	Java UUID

Path	Type	Description
`logstash`	`String`	ISO 8601 date with TimeZone UTC in format "yyyy-MM-dd'T'HH:mm:ssZ" e.g. 2017-04-01T00:00:00Z
`description`	`String`	Description or name
`dimension`	`String`	Name of a dimension in a matrix. E.g. distance, time, x, y, z, ...
`_links`	`Object`	Links to other resources

Example request

```
$ curl 'http://skylar.livingfire.de/api/settings/4' -i -X GET
```

Example response

```
HTTP/1.1 200 OK
Content-Type: application/hal+json;charset=UTF-8
Content-Length: 475

{
  "uuid" : "85f4fb14-8493-4ceb-945d-cdbd4f239cb9",
  "logstash" : "2017-04-01T00:00:00+02:00",
  "description" : "de",
  "dimension" : "ttsLanguage",
  "_links" : {
    "self" : {
      "href" : "http://skylar.livingfire.de/api/settings/4"
    },
    "setting" : {
      "href" : "http://skylar.livingfire.de/api/settings/4"
    },
    "relationDimensionHasDimension" : {
      "href" : "http://skylar.livingfire.de/api/settings/4/relationDimensionHasDimension"
    }
  }
}
```

Links

Relation	Description
`self`	Canonical link for this resource
`setting`	This Setting
`relationDimensionHasDimension`	Relation Dimension

Update

A `PATCH` request will update a Setting.

Example request

```
$ curl 'http://skylar.livingfire.de/api/settings/4' -i -X PATCH \
    -H 'Content-Type: application/hal+json' \
    -d '{
  "description" : "en"
}'
```

Example response

```
HTTP/1.1 204 No Content
```

Delete

A `DELETE` request will delete a Setting.

Example request

```
$ curl 'http://skylar.livingfire.de/api/settings/4' -i -X DELETE \
    -H 'Content-Type: application/hal+json'
```

Example response

```
HTTP/1.1 204 No Content
```

Search

Details

See Application-Level Profile Semantics (ALPS) [http://docs.spring.io/spring-data/rest/docs/current/reference/html/#metadata.alps]

Example request

```
$ curl 'http://skylar.livingfire.de/api/settings/search' -i -X GET
```

Example response

```
HTTP/1.1 200 OK
Content-Type: application/hal+json;charset=UTF-8
Content-Length: 555

{
  "_links" : {
    "findByDimension" : {
      "href" : "http://skylar.livingfire.de/api/settings/search/findByDimension{?dimension}",
      "templated" : true
    },
    "findByUuid" : {
      "href" : "http://skylar.livingfire.de/api/settings/search/findByUuid{?uuid}",
      "templated" : true
    },
    "findByDescription" : {
      "href" : "http://skylar.livingfire.de/api/settings/search/findByDescription{?description}",
      "templated" : true
    },
    "self" : {
      "href" : "http://skylar.livingfire.de/api/settings/search"
    }
  }
}
```

findByUuid

A `GET` request will retrieve a Setting.

Request parameters

Parameter	Description
`uuid`	Java UUID

Response fields

Path	Type	Description
`uuid`	`String`	Java UUID
`logstash`	`String`	ISO 8601 date with TimeZone UTC in format "yyyy-MM-

Path	Type	Description
		dd'T'HH:mm:ssZ" e.g. 2017-04-01T00:00:00Z
`description`	`String`	Description or name
`dimension`	`String`	Name of a dimension in a matrix. E.g. distance, time, x, y, z, …
`_links`	`Object`	Links to other resources

Example request

```
$ curl 'http://skylar.livingfire.de/api/settings/search/findByUuid?uuid=85f4fb14-8493-4ceb-945d-cdbd4f239cb9' -i -X GET
```

Example response

```
HTTP/1.1 200 OK
Content-Type: application/hal+json;charset=UTF-8
Content-Length: 475

{
  "uuid" : "85f4fb14-8493-4ceb-945d-cdbd4f239cb9",
  "logstash" : "2017-04-01T00:00:00+02:00",
  "description" : "en",
  "dimension" : "ttsLanguage",
  "_links" : {
    "self" : {
      "href" : "http://skylar.livingfire.de/api/settings/4"
    },
    "setting" : {
      "href" : "http://skylar.livingfire.de/api/settings/4"
    },
    "relationDimensionHasDimension" : {
      "href" : "http://skylar.livingfire.de/api/settings/4/relationDimensionHasDimension"
    }
  }
}
```

Links

Relation	Description
`self`	Canonical link for this resource
`setting`	This Setting
`relationDimensionHasDimension`	Relation Dimension

findByDescription

A `GET` request will retrieve a array of Setting.

Request parameters

Parameter	Description
`description`	Description or name

Response fields

Path	Type	Description
`_embedded.settings`	`Array`	An array of Setting
`_links`	`Object`	Links to other resources

Example request

```
$ curl 'http://skylar.livingfire.de/api/settings/search/findByDescription?description=de' -i -X GET
```

Example response

```
HTTP/1.1 200 OK
Content-Type: application/hal+json;charset=UTF-8
Content-Length: 170

{
  "_embedded" : {
    "settings" : [ ]
  },
  "_links" : {
    "self" : {
      "href" : "http://skylar.livingfire.de/api/settings/search/findByDescription"
    }
  }
}
```

Links

Relation	Description
`self`	Canonical link for this resource

Settings

Listing

A `GET` request will retrieve a paginated view [http://docs.spring.io/spring-data/rest/docs/current/reference/html/#paging-and-sorting] of all Setting.

Response fields

Path	Type	Description
`_embedded.settings`	`Array`	An array of Setting
`_links`	`Object`	Links to other resources
`page`	`Object`	paging information

Example request

```
$ curl 'http://skylar.livingfire.de/api/settings' -i -X GET
```

Example response

```
HTTP/1.1 200 OK
Content-Type: application/hal+json;charset=UTF-8
Content-Length: 1007

{
  "_embedded" : {
    "settings" : [ {
      "uuid" : "85f4fb14-8493-4ceb-945d-cdbd4f239cb9",
      "logstash" : "2017-04-01T00:00:00+02:00",
      "description" : "en",
      "dimension" : "ttsLanguage",
      "_links" : {
        "self" : {
          "href" : "http://skylar.livingfire.de/api/settings/4"
        },
        "setting" : {
          "href" : "http://skylar.livingfire.de/api/settings/4"
```

```
        },
        "relationDimensionHasDimension" : {
          "href" : "http://skylar.livingfire.de/api/settings/4/relationDimensionHasDimension"
        }
      }
    } ]
  },
  "_links" : {
    "self" : {
      "href" : "http://skylar.livingfire.de/api/settings{?page,size,sort}",
      "templated" : true
    },
    "profile" : {
      "href" : "http://skylar.livingfire.de/api/profile/settings"
    },
    "search" : {
      "href" : "http://skylar.livingfire.de/api/settings/search"
    }
  },
  "page" : {
    "size" : 20,
    "totalElements" : 1,
    "totalPages" : 1,
    "number" : 0
  }
}
```

Links

Relation	Description
`self`	Canonical link for this resource
`profile`	The Application-Level Profile Semantics (ALPS)
`search`	Canonical link to search resources

Technology

Details

See Application-Level Profile Semantics (ALPS) [http://docs.spring.io/spring-data/rest/docs/current/reference/html/#metadata.alps]

Example request

```
$ curl 'http://skylar.livingfire.de/api/profile/technologys' -i -X GET
```

Create

A `POST` request will create a Technology.

Example request

```
$ curl 'http://skylar.livingfire.de/api/technologys' -i -X POST \
    -H 'Content-Type: application/hal+json' \
    -d '{
  "id" : null,
  "uuid" : "f3ad6eb6-8f0d-44a5-a936-9f78d030f885",
  "logstash" : "2017-04-01T00:00:00Z",
  "description" : "Java"
}'
```

Example response

```
HTTP/1.1 201 Created
Location: http://skylar.livingfire.de/api/technologys/5
```

Retrieve

A `GET` request will retrieve a Technology.

Response fields

Path	Type	Description
`uuid`	`String`	Java UUID
`logstash`	`String`	ISO 8601 date with TimeZone UTC in format "yyyy-MM-dd'T'HH:mm:ssZ" e.g. 2017-04-01T00:00:00Z
`description`	`String`	Description or name
`_links`	`Object`	Links to other resources

Example request

```
$ curl 'http://skylar.livingfire.de/api/technologys/5' -i -X GET
```

Example response

```
HTTP/1.1 200 OK
Content-Type: application/hal+json;charset=UTF-8
Content-Length: 313

{
  "uuid" : "f3ad6eb6-8f0d-44a5-a936-9f78d030f885",
  "logstash" : "2017-04-01T00:00:00Z",
  "description" : "Java",
  "_links" : {
    "self" : {
      "href" : "http://skylar.livingfire.de/api/technologys/5"
    },
    "technology" : {
      "href" : "http://skylar.livingfire.de/api/technologys/5"
    }
  }
}
```

Links

Relation	Description
`self`	Canonical link for this resource
`technology`	This Technology

Update

A `PATCH` request will update a Technology.

Example request

```
$ curl 'http://skylar.livingfire.de/api/technologys/5' -i -X PATCH \
    -H 'Content-Type: application/hal+json' \
    -d '{
  "description" : "Java"
}'
```

Example response

```
HTTP/1.1 204 No Content
```

Delete

A `DELETE` request will delete a Technology.

Example request

```
$ curl 'http://skylar.livingfire.de/api/technologys/5' -i -X DELETE \
    -H 'Content-Type: application/hal+json'
```

Example response

```
HTTP/1.1 204 No Content
```

Search

Details

See Application-Level Profile Semantics (ALPS) [http://docs.spring.io/spring-data/rest/docs/current/reference/html/#metadata.alps]

Example request

```
$ curl 'http://skylar.livingfire.de/api/technologys/search' -i -X GET
```

Example response

```
HTTP/1.1 200 OK
Content-Type: application/hal+json;charset=UTF-8
Content-Length: 412

{
  "_links" : {
    "findByDescription" : {
      "href" : "http://skylar.livingfire.de/api/technologys/search/findByDescription{?description}",
      "templated" : true
    },
    "findByUuid" : {
      "href" : "http://skylar.livingfire.de/api/technologys/search/findByUuid{?uuid}",
      "templated" : true
    },
    "self" : {
      "href" : "http://skylar.livingfire.de/api/technologys/search"
    }
  }
}
```

findByUuid

A `GET` request will retrieve a Technology.

Request parameters

Parameter	Description
`uuid`	Java UUID

Response fields

Path	Type	Description
`uuid`	`String`	Java UUID
`logstash`	`String`	ISO 8601 date with TimeZone UTC in format "yyyy-MM-dd'T'HH:mm:ssZ" e.g. 2017-04-01T00:00:00Z

Path	Type	Description
`description`	`String`	Description or name
`_links`	`Object`	Links to other resources

Example request

```
$ curl 'http://skylar.livingfire.de/api/technologys/search/findByUuid?uuid=f3ad6eb6-8f0d-44a5-a936-9f78d030f885' -i -X GET
```

Example response

```
HTTP/1.1 200 OK
Content-Type: application/hal+json;charset=UTF-8
Content-Length: 313

{
  "uuid" : "f3ad6eb6-8f0d-44a5-a936-9f78d030f885",
  "logstash" : "2017-04-01T00:00:00Z",
  "description" : "Java",
  "_links" : {
    "self" : {
      "href" : "http://skylar.livingfire.de/api/technologys/5"
    },
    "technology" : {
      "href" : "http://skylar.livingfire.de/api/technologys/5"
    }
  }
}
```

Links

Relation	Description
`self`	Canonical link for this resource
`technology`	This Technology

findByDescription

A `GET` request will retrieve a array of Technology.

Request parameters

Parameter	Description
`description`	Description or name

Response fields

Path	Type	Description
`_embedded.technologys`	`Array`	An array of Technology
`_links`	`Object`	Links to other resources

Example request

```
$ curl 'http://skylar.livingfire.de/api/technologys/search/findByDescription?description=Java' -i -X GET
```

Example response

```
HTTP/1.1 200 OK
Content-Type: application/hal+json;charset=UTF-8
Content-Length: 538

{
  "_embedded" : {
```

```
    "technologys" : [ {
      "uuid" : "f3ad6eb6-8f0d-44a5-a936-9f78d030f885",
      "logstash" : "2017-04-01T00:00:00Z",
      "description" : "Java",
      "_links" : {
        "self" : {
          "href" : "http://skylar.livingfire.de/api/technologys/5"
        },
        "technology" : {
          "href" : "http://skylar.livingfire.de/api/technologys/5"
        }
      }
    } ]
  },
  "_links" : {
    "self" : {
      "href" : "http://skylar.livingfire.de/api/technologys/search/findByDescription"
    }
  }
}
```

Links

Relation	Description
`self`	Canonical link for this resource

Technologies

Listing

A `GET` request will retrieve a paginated view [http://docs.spring.io/spring-data/rest/docs/current/reference/html/#paging-and-sorting] of all Technology.

Response fields

Path	Type	Description
`_embedded.technologys`	`Array`	An array of Technology
`_links`	`Object`	Links to other resources
`page`	`Object`	paging information

Example request

```
$ curl 'http://skylar.livingfire.de/api/technologys' -i -X GET
```

Example response

```
HTTP/1.1 200 OK
Content-Type: application/hal+json;charset=UTF-8
Content-Length: 841

{
  "_embedded" : {
    "technologys" : [ {
      "uuid" : "f3ad6eb6-8f0d-44a5-a936-9f78d030f885",
      "logstash" : "2017-04-01T00:00:00Z",
      "description" : "Java",
      "_links" : {
        "self" : {
          "href" : "http://skylar.livingfire.de/api/technologys/5"
        },
        "technology" : {
          "href" : "http://skylar.livingfire.de/api/technologys/5"
        }
      }
    } ]
  },
  "_links" : {
```

```
    "self" : {
      "href" : "http://skylar.livingfire.de/api/technologys{?page,size,sort}",
      "templated" : true
    },
    "profile" : {
      "href" : "http://skylar.livingfire.de/api/profile/technologys"
    },
    "search" : {
      "href" : "http://skylar.livingfire.de/api/technologys/search"
    }
  },
  "page" : {
    "size" : 20,
    "totalElements" : 1,
    "totalPages" : 1,
    "number" : 0
  }
}
```

Links

Relation	Description
`self`	Canonical link for this resource
`profile`	The Application-Level Profile Semantics (ALPS)
`search`	Canonical link to search resources

User

The information stored for a Skylar User Account.

Details

See Application-Level Profile Semantics (ALPS) [http://docs.spring.io/spring-data/rest/docs/current/reference/html/#metadata.alps]

Example request

```
$ curl 'http://skylar.livingfire.de/api/profile/users' -i -X GET
```

Create

A `POST` request will create a User.

Example request

```
$ curl 'http://skylar.livingfire.de/api/users' -i -X POST \
    -H 'Content-Type: application/hal+json' \
    -d '{
  "id" : null,
  "uuid" : "9ea65c1c-ae99-4a1a-b840-e8d7f5ec2c10",
  "logstash" : "2017-04-01T00:00:00+02:00",
  "description" : "Thomas",
  "relationSettingsSetBy" : [ ],
  "relationListdatasCanView" : [ ],
  "locationConfiguration" : {
    "language" : "de",
    "region" : "DE",
    "timeZoneString" : "Europe/Berlin",
    "locale" : "de_DE",
    "zoneId" : "Europe/Berlin",
    "zonedDateTime" : "2019-01-28T16:08:13.091+01:00"
  },
  "ttsConfiguration" : {
    "ttsGender" : "female",
    "ttsLanguage" : "de",
    "location" : {
```

```
      "language" : "de",
      "region" : "DE",
      "timeZoneString" : "Europe/Berlin",
      "locale" : "de_DE",
      "zoneId" : "Europe/Berlin",
      "zonedDateTime" : "2019-01-28T16:08:13.093+01:00"
    }
  }
}'
```

Example response

```
HTTP/1.1 201 Created
Location: http://skylar.livingfire.de/api/users/6
```

Retrieve

A `GET` request will retrieve a User.

Response fields

Path	Type	Description
`uuid`	`String`	Java UUID
`logstash`	`String`	ISO 8601 date with TimeZone UTC in format "yyyy-MM-dd'T'HH:mm:ssZ" e.g. 2017-04-01T00:00:00Z
`description`	`String`	Description or name
`_links`	`Object`	Links to other resources
`locationConfiguration.language`	`String`	LocationConfiguration - language e.g. de
`locationConfiguration.region`	`String`	LocationConfiguration - region e.g. DE
`locationConfiguration.timeZoneString`	`String`	LocationConfiguration - timeZoneString e.g. Europe/Berlin
`locationConfiguration.locale`	`String`	LocationConfiguration - locale e.g. de_DE
`locationConfiguration.zoneId`	`String`	LocationConfiguration - zoneId e.g. Europe/Berlin
`locationConfiguration.zonedDateTime`	`String`	LocationConfiguration - zonedDateTime e.g. 2017-11-02T19:24:13.514+01:00
`ttsConfiguration.`	`Object`	TtsConfiguration - e.g.
`ttsConfiguration.ttsGender`	`String`	TtsConfiguration ttsGender - e.g. female
`ttsConfiguration.ttsLanguage`	`String`	TtsConfiguration ttsLanguage - e.g. german
`ttsConfiguration.locationLanguage`	`String`	LocationConfiguration - language e.g. de

Path	Type	Description
`ttsConfiguration.location.region`	String	LocationConfiguration - region e.g. DE
`ttsConfiguration.location.timeZoneString`	String	LocationConfiguration - timeZoneString e.g. Europe/Berlin
`ttsConfiguration.location.locale`	String	LocationConfiguration - locale e.g. de_DE
`ttsConfiguration.location.zoneId`	String	LocationConfiguration - zoneId e.g. Europe/Berlin
`ttsConfiguration.location.zonedDateTime`	String	LocationConfiguration - zonedDateTime e.g. 2017-11-02T19:24:13.514+01:00

Example request

```
$ curl 'http://skylar.livingfire.de/api/users/6' -i -X GET
```

Example response

```
HTTP/1.1 200 OK
Content-Type: application/hal+json;charset=UTF-8
Content-Length: 1088

{
  "uuid" : "9ea65c1c-ae99-4a1a-b840-e8d7f5ec2c10",
  "logstash" : "2017-04-01T00:00:00+02:00",
  "description" : "Thomas",
  "locationConfiguration" : {
    "language" : "de",
    "region" : "DE",
    "timeZoneString" : "Europe/Berlin",
    "locale" : "de_DE",
    "zoneId" : "Europe/Berlin",
    "zonedDateTime" : "2019-01-28T16:08:13.188+01:00"
  },
  "ttsConfiguration" : {
    "ttsGender" : "female",
    "ttsLanguage" : "de",
    "location" : {
      "language" : "de",
      "region" : "DE",
      "timeZoneString" : "Europe/Berlin",
      "locale" : "de_DE",
      "zoneId" : "Europe/Berlin",
      "zonedDateTime" : "2019-01-28T16:08:13.192+01:00"
    }
  },
  "_links" : {
    "self" : {
      "href" : "http://skylar.livingfire.de/api/users/6"
    },
    "user" : {
      "href" : "http://skylar.livingfire.de/api/users/6"
    },
    "relationListdatasCanView" : {
      "href" : "http://skylar.livingfire.de/api/users/6/relationListdatasCanView"
    },
    "relationSettingsSetBy" : {
      "href" : "http://skylar.livingfire.de/api/users/6/relationSettingsSetBy"
    }
  }
}
```

Links

Relation	Description
`self`	Canonical link for this resource

Relation	Description
`user`	This User
`relationListdatasCanView`	Relation ListDatas
`relationSettingsSetBy`	Relation Settings

Update

A `PATCH` request will update a User.

Example request

```
$ curl 'http://skylar.livingfire.de/api/users/6' -i -X PATCH \
    -H 'Content-Type: application/hal+json' \
    -d '{
  "description" : "Skylar"
}'
```

Example response

```
HTTP/1.1 204 No Content
```

Delete

A `DELETE` request will delete a User.

Example request

```
$ curl 'http://skylar.livingfire.de/api/users/6' -i -X DELETE \
    -H 'Content-Type: application/hal+json'
```

Example response

```
HTTP/1.1 204 No Content
```

Search

Details

See Application-Level Profile Semantics (ALPS) [http://docs.spring.io/spring-data/rest/docs/current/reference/html/#metadata.alps]

Example request

```
$ curl 'http://skylar.livingfire.de/api/users/search' -i -X GET
```

Example response

```
HTTP/1.1 200 OK
Content-Type: application/hal+json;charset=UTF-8
Content-Length: 521

{
  "_links" : {
    "findByUuid" : {
      "href" : "http://skylar.livingfire.de/api/users/search/findByUuid{?uuid}",
      "templated" : true
    },
    "findByDescription" : {
      "href" : "http://skylar.livingfire.de/api/users/search/findByDescription{?description}",
      "templated" : true
    },
    "exportAllRelationsToCsv" : {
      "href" : "http://skylar.livingfire.de/api/users/search/exportAllRelationsToCsv"
```

```
    },
    "self" : {
      "href" : "http://skylar.livingfire.de/api/users/search"
    }
  }
}
```

findByUuid

A `GET` request will retrieve a User.

Request parameters

Parameter	Description
`uuid`	Java UUID

Response fields

Path	Type	Description
`uuid`	`String`	Java UUID
`logstash`	`String`	ISO 8601 date with TimeZone UTC in format "yyyy-MM-dd'T'HH:mm:ssZ" e.g. 2017-04-01T00:00:00Z
`description`	`String`	Description or name
`_links`	`Object`	Links to other resources
`locationConfiguration.language`	`String`	LocationConfiguration - language e.g. de
`locationConfiguration.region`	`String`	LocationConfiguration - region e.g. DE
`locationConfiguration.timeZoneString`	`String`	LocationConfiguration - timeZoneString e.g. Europe/Berlin
`locationConfiguration.locale`	`String`	LocationConfiguration - locale e.g. de_DE
`locationConfiguration.zoneId`	`String`	LocationConfiguration - zoneId e.g. Europe/Berlin
`locationConfiguration.zonedDateTime`	`String`	LocationConfiguration - zonedDateTime e.g. 2017-11-02T19:24:13.514+01:00
`ttsConfiguration.`	`Object`	TtsConfiguration - e.g.
`ttsConfiguration.ttsGender`	`String`	TtsConfiguration ttsGender - e.g. female
`ttsConfiguration.ttsLanguage`	`String`	TtsConfiguration ttsLanguage - e.g. german
`ttsConfiguration.locationLanguage`	`String`	LocationConfiguration - language e.g. de

Path	Type	Description
ttsConfiguration.location.region	String	LocationConfiguration - region e.g. DE
ttsConfiguration.location.timeZoneString	String	LocationConfiguration - timeZoneString e.g. Europe/Berlin
ttsConfiguration.location.locale	String	LocationConfiguration - locale e.g. de_DE
ttsConfiguration.location.zoneId	String	LocationConfiguration - zoneId e.g. Europe/Berlin
ttsConfiguration.location.zonedDateTime	String	LocationConfiguration - zonedDateTime e.g. 2017-11-02T19:24:13.514+01:00

Example request

```
$ curl 'http://skylar.livingfire.de/api/users/search/findByUuid?uuid=9ea65c1c-ae99-4a1a-b840-e8d7f5ec2c10' -i -X GET
```

Example response

```
HTTP/1.1 200 OK
Content-Type: application/hal+json;charset=UTF-8
Content-Length: 1088

{
  "uuid" : "9ea65c1c-ae99-4a1a-b840-e8d7f5ec2c10",
  "logstash" : "2017-04-01T00:00:00+02:00",
  "description" : "Skylar",
  "locationConfiguration" : {
    "language" : "de",
    "region" : "DE",
    "timeZoneString" : "Europe/Berlin",
    "locale" : "de_DE",
    "zoneId" : "Europe/Berlin",
    "zonedDateTime" : "2019-01-28T16:08:13.386+01:00"
  },
  "ttsConfiguration" : {
    "ttsGender" : "female",
    "ttsLanguage" : "de",
    "location" : {
      "language" : "de",
      "region" : "DE",
      "timeZoneString" : "Europe/Berlin",
      "locale" : "de_DE",
      "zoneId" : "Europe/Berlin",
      "zonedDateTime" : "2019-01-28T16:08:13.387+01:00"
    }
  },
  "_links" : {
    "self" : {
      "href" : "http://skylar.livingfire.de/api/users/6"
    },
    "user" : {
      "href" : "http://skylar.livingfire.de/api/users/6"
    },
    "relationListdatasCanView" : {
      "href" : "http://skylar.livingfire.de/api/users/6/relationListdatasCanView"
    },
    "relationSettingsSetBy" : {
      "href" : "http://skylar.livingfire.de/api/users/6/relationSettingsSetBy"
    }
  }
}
```

Links

Relation	Description
self	Canonical link for this resource

Relation	Description
`user`	This User
`relationListdatasCanView`	Relation ListDatas
`relationSettingsSetBy`	Relation Settings

findByDescription

A `GET` request will retrieve a array of User.

Request parameters

Parameter	Description
`description`	Description or name

Response fields

Path	Type	Description
`_embedded.users`	`Array`	An array of User
`_links`	`Object`	Links to other resources

Example request

```
$ curl 'http://skylar.livingfire.de/api/users/search/findByDescription?description=Thomas' -i -X GET
```

Example response

```
HTTP/1.1 200 OK
Content-Type: application/hal+json;charset=UTF-8
Content-Length: 164

{
  "_embedded" : {
    "users" : [ ]
  },
  "_links" : {
    "self" : {
      "href" : "http://skylar.livingfire.de/api/users/search/findByDescription"
    }
  }
}
```

Links

Relation	Description
`self`	Canonical link for this resource

Users

Listing

A `GET` request will retrieve a paginated view [http://docs.spring.io/spring-data/rest/docs/current/reference/html/#paging-and-sorting] of all User.

Response fields

Path	Type	Description
`_embedded.users`	`Array`	An array of User
`_links`	`Object`	Links to other resources
`page`	`Object`	paging information

Example request

```
$ curl 'http://skylar.livingfire.de/api/users' -i -X GET
```

Example response

```
HTTP/1.1 200 OK
Content-Type: application/hal+json;charset=UTF-8
Content-Length: 1696

{
  "_embedded" : {
    "users" : [ {
      "uuid" : "9ea65c1c-ae99-4a1a-b840-e8d7f5ec2c10",
      "logstash" : "2017-04-01T00:00:00+02:00",
      "description" : "Skylar",
      "locationConfiguration" : {
        "language" : "de",
        "region" : "DE",
        "timeZoneString" : "Europe/Berlin",
        "locale" : "de_DE",
        "zoneId" : "Europe/Berlin",
        "zonedDateTime" : "2019-01-28T16:08:13.336+01:00"
      },
      "ttsConfiguration" : {
        "ttsGender" : "female",
        "ttsLanguage" : "de",
        "location" : {
          "language" : "de",
          "region" : "DE",
          "timeZoneString" : "Europe/Berlin",
          "locale" : "de_DE",
          "zoneId" : "Europe/Berlin",
          "zonedDateTime" : "2019-01-28T16:08:13.336+01:00"
        }
      },
      "_links" : {
        "self" : {
          "href" : "http://skylar.livingfire.de/api/users/6"
        },
        "user" : {
          "href" : "http://skylar.livingfire.de/api/users/6"
        },
        "relationListdatasCanView" : {
          "href" : "http://skylar.livingfire.de/api/users/6/relationListdatasCanView"
        },
        "relationSettingsSetBy" : {
          "href" : "http://skylar.livingfire.de/api/users/6/relationSettingsSetBy"
        }
      }
    } ]
  },
  "_links" : {
    "self" : {
      "href" : "http://skylar.livingfire.de/api/users{?page,size,sort}",
      "templated" : true
    },
    "profile" : {
      "href" : "http://skylar.livingfire.de/api/profile/users"
    },
    "search" : {
      "href" : "http://skylar.livingfire.de/api/users/search"
    }
  },
  "page" : {
    "size" : 20,
    "totalElements" : 1,
    "totalPages" : 1,
    "number" : 0
  }
}
```

Links

Relation	Description
`self`	Canonical link for this resource
`profile`	The Application-Level Profile Semantics (ALPS)
`search`	Canonical link to search resources

Export

A `GET` request will list the index.

Example request

```
$ curl -i http://skylar.livingfire.de/api/export
```

Response fields

Path	Type	Description
`_links`	`Object`	Links to other resources

Links

Relation	Description
`self`	Canonical link for this resource
`exportDatabase`	See Export Database

Database

A `GET` request will download the database as a ZIP file.

Example request

```
$ curl -D headers.txt -o skylar_database.zip http://skylar.livingfire.de/api/export/database
$ cat headers.txt
```

Response Header

```
HTTP/1.1 201
Content-Disposition: attachment; filename="20160401_1706_skylar_database.zip"
```

Import

A `GET` request will list the index.

Example request

```
$ curl -i http://skylar.livingfire.de/api/import
```

Response fields

Path	Type	Description
`_links`	`Object`	Links to other resources

Links

Relation	Description
`self`	Canonical link for this resource
`importOruxmaps`	See Oruxmaps
`importLibra`	See Libra
`importMicrolife`	See Microlife
`importGarmin`	See Garmin

Oruxmaps

A `POST` request will import a Oruxmaps [http://www.oruxmaps.com] KML [https://developers.google.com/kml/] file.

Example request

```
$ curl -i --form file=@oruxmaps.kml http://skylar.livingfire.de/api/import/oruxmaps
```

Example response

```
HTTP/1.1 201 OK
Content-Type: application/json;charset=UTF-8

{
  "status" : 201,
  "code" : "009004",
  "userMessage" : "upload oruxmaps successful: oruxmaps.kml",
  "developerMessage" : "upload oruxmaps successful",
  "moreInfoURL" : "https://github.com/phoen1x/skylar-the-scholar/blob/master/book_en.pdf"
}
```

Libra

A `POST` request will import a Libra [https://play.google.com/store/apps/details?id=net.cachapa.libra] CSV file.

Example request

```
$ echo '#Version:5
#Units:kg

#date;weight;weight trend;body fat;body fat trend;comment
2017-04-01 00:00:00;75.1;75.1;;;' \
    > libra.csv

$ curl -i --form file=@libra.csv http://skylar.livingfire.de/api/import/libra
```

Example response

```
HTTP/1.1 201 OK
Content-Type: application/json;charset=UTF-8

{
  "status" : 201,
  "code" : "009007",
  "userMessage" : "libra import successful: libra.csv",
  "developerMessage" : "libra import successful",
  "moreInfoURL" : "https://github.com/phoen1x/skylar-the-scholar/blob/master/book_en.pdf"
```

```
}
```

Microlife

A `POST` request will import a Microlife [https://play.google.com/store/apps/details?id=microlife.a6p2.bluetooth.app] CSV file.

Example request

```
$ echo 'Name : Skylar
ID : skylar
Geschlecht : Weiblich
Geburtsdatum : 1. April 2010
Datum,Zeit,Systole,Diastole,Puls,Arrhythmie,MAM
"01.04.17","21:37","120","80","70","",""' \
    > microlife.csv

$ curl -i --form file=@microlife.csv http://skylar.livingfire.de/api/import/microlife
```

Example response

```
HTTP/1.1 201 OK
Content-Type: application/json;charset=UTF-8

{
  "status" : 201,
  "code" : "009009",
  "userMessage" : "microlife import successful: microlife.csv",
  "developerMessage" : "microlife import successful",
  "moreInfoURL" : "https://github.com/phoen1x/skylar-the-scholar/blob/master/book_en.pdf"
}
```

Garmin

A `GET` request will list the index.

Example request

```
$ curl -i http://skylar.livingfire.de/api/import/garmin
```

Response fields

Path	Type	Description
`_links`	`Object`	Links to other resources

Links

Relation	Description
`self`	Canonical link for this resource
`importGarminActivity`	See Garmin Activity

Activity

A `POST` request will import a Training Center XML (TCX) [https://en.wikipedia.org/wiki/Training_Center_XML] file.

Example request

```
$ # download sample https://developer.garmin.com/garmin-connect-api/sample-data/
$ curl https://developer.garmin.com/downloads/connect-api/sample_file.tcx > activity.tcx
$ curl -i --form file=@activity.tcx http://skylar.livingfire.de/api/import/garmin/activity
```

Example response

```
HTTP/1.1 201 OK
Content-Type: application/json;charset=UTF-8
```

```
{
  "status" : 201,
  "code" : "009005",
  "userMessage" : "garmin activity import successful: activity.tcx",
  "developerMessage" : "garmin activity import successful",
  "moreInfoURL" : "https://github.com/phoen1x/skylar-the-scholar/blob/master/book_en.pdf"
}
```

JavaScript

Helper methods for JavaScript. A `GET` request will list the index.

Example request

```
$ curl -i http://skylar.livingfire.de/api/javascript
```

Response fields

Path	Type	Description
`_links`	`Object`	Links to other resources

Links

Relation	Description
`self`	Canonical link for this resource
`entityProperties`	See Entity properties
`ttsGreeting`	See TTS greeting
`ttsKanboardOverdue`	See Kanboard overdue
`emailKanboardShopping`	See Kanboard shopping email
`ttsKanboardShopping`	See Kanboard shopping TTS
`ttsWeather`	See TTS Weather
`ttsGoogleCalendar`	See TTS Google Calendar
`listdataGroupDimensions`	See Listdata Group-Dimension Mapping
`notesOfActiveFirms`	See Notes of active Firms

Entity properties

A `GET` request will retrieve properties needed to create a new Skylar entity.

Response fields

Path	Type	Description
`uuid`	`String`	Java UUID
`logstash`	`String`	ISO 8601 date with TimeZone UTC in format "yyyy-MM-dd'T'HH:mm:ssZ" e.g. 2017-04-01T00:00:00Z

Example request

```
$ curl 'http://skylar.livingfire.de/api/javascript/entityProperties' -i -X GET
```

Example response

```
HTTP/1.1 200 OK
Content-Type: application/json;charset=UTF-8
Content-Length: 86

{"uuid":"79e8bdea-da15-4869-be5f-c31296f16c6e","logstash":"2017-07-01T22:44:25+02:00"}
```

TTS Greeting

A `GET` will trigger a TTS greeting.

Response fields

Path	Type	Description
`status`	`Number`	a HTTP status code like 201
`code`	`String`	native Skylar code like 005001 see SkylarCodeConstant.java
`userMessage`	`String`	Message shown in the user interface
`developerMessage`	`String`	Additional information for debugging

Example request

```
$ curl 'http://skylar.livingfire.de/api/javascript/ttsGreeting' -i -X GET
```

Example response

```
HTTP/1.1 200 OK
Content-Type: application/json;charset=UTF-8
Content-Length: 118

{"status":201,"code":"005001","userMessage":"Text-To-Speech created","developerMessage":"message sent into JMS queue"}
```

Kanboard overdue

A `GET` will trigger a TTS output of all overdue kanboard tasks in Skylar.

Response fields

Path	Type	Description
`status`	`Number`	a HTTP status code like 201
`code`	`String`	native Skylar code like 005001 see SkylarCodeConstant.java
`userMessage`	`String`	Message shown in the user interface
`developerMessage`	`String`	Additional information for debugging

Example request

```
$ curl 'http://skylar.livingfire.de/api/javascript/ttsKanboardOverdue' -i -X GET
```

Example response

```
HTTP/1.1 200 OK
Content-Type: application/json;charset=UTF-8
Content-Length: 118

{"status":201,"code":"005001","userMessage":"Text-To-Speech created","developerMessage":"message sent into JMS queue"}
```

Kanboard shopping TTS

A `GET` will trigger a TTS output with the shopping list.

Response fields

Path	Type	Description
`status`	`Number`	a HTTP status code like 201
`code`	`String`	native Skylar code like 005001 see SkylarCodeConstant.java
`userMessage`	`String`	Message shown in the user interface
`developerMessage`	`String`	Additional information for debugging

Example request

```
$ curl 'http://skylar.livingfire.de/api/javascript/ttsKanbaordShopping' -i -X GET
```

Example response

```
HTTP/1.1 200 OK
Content-Type: application/json;charset=UTF-8
Content-Length: 118

{"status":201,"code":"005001","userMessage":"Text-To-Speech created","developerMessage":"message sent into JMS queue"}
```

Kanboard shopping email

A `GET` will trigger a Email with the shopping list.

Response fields

Path	Type	Description
`status`	`Number`	a HTTP status code like 201
`code`	`String`	native Skylar code like 005001 see SkylarCodeConstant.java
`userMessage`	`String`	Message shown in the user interface
`developerMessage`	`String`	Additional information for debugging

Example request

```
$ curl 'http://skylar.livingfire.de/api/javascript/emailKanbaordShopping' -i -X GET
```

Example response

```
HTTP/1.1 200 OK
Content-Type: application/json;charset=UTF-8
Content-Length: 95

{"status":201,"code":"004001","userMessage":"email sent OK","developerMessage":"email sent OK"}
```

TTS Weather

A `GET` will trigger a TTS output of a weather report.

Response fields

Path	Type	Description
`status`	`Number`	a HTTP status code like 201
`code`	`String`	native Skylar code like 005001 see SkylarCodeConstant.java
`userMessage`	`String`	Message shown in the user interface
`developerMessage`	`String`	Additional information for debugging

Example request

```
$ curl 'http://skylar.livingfire.de/api/javascript/ttsWeather' -i -X GET
```

Example response

```
HTTP/1.1 200 OK
Content-Type: application/json;charset=UTF-8
Content-Length: 118

{"status":201,"code":"005001","userMessage":"Text-To-Speech created","developerMessage":"message sent into JMS queue"}
```

TTS Google Calendar

A `GET` will trigger a TTS output of a Google Calendar events overdue report.

Response fields

Path	Type	Description
`status`	`Number`	a HTTP status code like 201
`code`	`String`	native Skylar code like 005001 see SkylarCodeConstant.java
`userMessage`	`String`	Message shown in the user interface
`developerMessage`	`String`	Additional information for debugging

Example request

```
$ curl 'http://skylar.livingfire.de/api/javascript/ttsGoogleCalendar' -i -X GET
```

Example response

```
HTTP/1.1 200 OK
Content-Type: application/json;charset=UTF-8
Content-Length: 118

{"status":201,"code":"005001","userMessage":"Text-To-Speech created","developerMessage":"message sent into JMS queue"}
```

Listdata Group-Dimension Mapping

A `GET` will retrieve a mapping of all dimensions to their coresponding groups. Group and dimension in this context mean the properties: Listdata.group and Listdata.dimension

Response fields

Path	Type	Description
`[].group`	`String`	group
`[].dimension`	`String`	dimension

Example request

```
$ curl 'http://skylar.livingfire.de/api/javascript/listdataGroupDimensions' -i -X GET
```

Example response

```
HTTP/1.1 200 OK
Content-Type: application/json;charset=UTF-8
Content-Length: 84

[{"group":"body","dimension":"bodyWeight"},{"group":"body","dimension":"heartrate"}]
```

Notes of Active Firms

A `GET` will retrieve a array of Strings with all Note.description of Firm.active == true. The data can be uses as an overview of all notes. (activity log)

Example request

```
$ curl 'http://skylar.livingfire.de/api/javascript/notesOfActiveFirms' -i -X GET
```

Example response

```
HTTP/1.1 200 OK
Content-Type: application/json;charset=UTF-8
Content-Length: 2

[]
```

Selenium

A `GET` request will list the index.

Example request

```
$ curl -i http://skylar.livingfire.de/api/selenium
```

Response fields

Path	Type	Description
`_links`	`Object`	Links to other resources

Links

Relation	Description
`self`	Canonical link for this resource
`seleniumWebpageOpen`	See web page open
`seleniumWebRadio`	See web radio
`seleniumSessionClose`	See session close
`seleniumSessionHtml`	See session html

web page open

A `GET` request will open a url in a web browser session.

Request parameters

Parameter	Description
`open`	a URL like https://www.youtube.com/

Response fields

Path	Type	Description
`status`	`Number`	a HTTP status code like 201
`code`	`String`	native Skylar code like 005001 see SkylarCodeConstant.java
`userMessage`	`String`	Message shown in the user interface
`developerMessage`	`String`	Additional information for debugging

Example request

```
$ curl 'http://skylar.livingfire.de/api/selenium/webpage?open=https://www.youtube.com/' -i -X GET
```

Example response

```
HTTP/1.1 200 OK
Content-Type: application/json;charset=UTF-8
Content-Length: 115

{"status":200,"code":"011001","userMessage":"selenium webpage opened","developerMessage":"selenium webpage opened"}
```

web radio

A `GET` request will start the web radio.

Response fields

Path	Type	Description
`status`	`Number`	a HTTP status code like 201
`code`	`String`	native Skylar code like 005001 see SkylarCodeConstant.java
`userMessage`	`String`	Message shown in the user interface
`developerMessage`	`String`	Additional information for debugging

Example request

```
$ curl 'http://skylar.livingfire.de/api/selenium/webRadio' -i -X GET
```

Example response

```
HTTP/1.1 200 OK
Content-Type: application/json;charset=UTF-8
Content-Length: 115

{"status":200,"code":"011001","userMessage":"selenium webpage opened","developerMessage":"selenium webpage opened"}
```

session close

A `GET` request will close the web browser session.

Response fields

Path	Type	Description
`status`	`Number`	a HTTP status code like 201
`code`	`String`	native Skylar code like 005001 see SkylarCodeConstant.java
`userMessage`	`String`	Message shown in the user interface
`developerMessage`	`String`	Additional information for debugging

Example request

```
$ curl 'http://skylar.livingfire.de/api/selenium/session/close' -i -X GET
```

Example response

```
HTTP/1.1 200 OK
Content-Type: application/json;charset=UTF-8
Content-Length: 115

{"status":200,"code":"011002","userMessage":"selenium session closed","developerMessage":"selenium session closed"}
```

session HTML

A `GET` request will get the HTML of the web browser session.

Example request

```
$ curl 'http://skylar.livingfire.de/api/selenium/session/html' -i -X GET
```

Example response

```
HTTP/1.1 200 OK
Content-Type: text/plain;charset=ISO-8859-1
Content-Length: 29

<html><body>foo</body></html>
```

Danksagung

Vielen Dank an alle die Software für dieses rießige Projekt bereitgestellt haben. Euch alle namentlich aufzuzählen ist wohl nicht möglich aber ich bemühe mich Quellen des Skylar Quellcodes zu benennen.

- flag-icon-css [http://flag-icon-css.lip.is/] on Github [https://github.com/pages-themes/dinky]
- Fontawesome [http://fontawesome.io/] on Github [https://github.com/lipis/flag-icon-css]
- Jekyll Dinky Theme [https://github.com/pages-themes/dinky] on Github [https://github.com/pages-themes/dinky]
- Kanboard - Simple and open source visual task board [http://kanboard.net] on Github [https://github.com/kanboard]
- rJava [https://www.rforge.net/rJava/] on Github [https://github.com/s-u/rJava]
- RNeo4j [https://github.com/nicolewhite/RNeo4j] on Github [https://github.com/nicolewhite/RNeo4j]
- Shields.io - Quality metadata badges for open source projects [http://shields.io] on Github [https://github.com/badges]
- The MARY Text-to-Speech System [http://mary.dfki.de] on Github [https://github.com/marytts]

Sollte Ihr Projekt fehlen oder falsch dargestellt worden sein senden Sie einfach einen pull-request [https://github.com/phoen1x/skylar-the-scholar] oder eröffnen Sie ein issue [https://github.com/phoen1x/skylar-the-scholar/issues].

Changelog

2.0.3 - Spring Boot 2.0.x

Mon Jan 28 15:33:09 CET 2019

- Upgrade Spring Boot to version 2.0.7
- Neo4j version 3.1.5

2.0.2 - Admin Channel

Sat Apr 21 18:33:37 CEST 2018

- Administrator can send emails with commands to Skylar

2.0.1 - Burglar alarm

Tue Apr 17 17:56:13 CEST 2018

- Check if IP addresses (e.g. mobile phone) is online
- TTS if IP is offline
- Send email with picture attached if IP is offline

2.0.0 - Spring Data Rest, React JS

Fri Apr 21 15:35:12 CEST 2017

- upgrade to Spring Boot 1.5.x
- Spring Data Rest [http://projects.spring.io/spring-data-rest/]
- Spring REST Docs [http://docs.spring.io/spring-restdocs/docs/current/reference/html5/]
- Spring WebSocket [https://spring.io/guides/gs/messaging-stomp-websocket/]
- Elasticsearch [https://www.elastic.co/products/elasticsearch] fulltext search
- ReactJS [https://facebook.github.io/react/] replaces AngularJS. Google breaks API with AngularJS2
- Electron Client [https://electron.atom.io/]
- Node.js [https://nodejs.org] overhaul in favour of create-react-app [https://github.com/facebookincubator/create-react-app]
- RabbitMQ [https://www.rabbitmq.com/] message broker
- StompJS [http://jmesnil.net/stomp-websocket/doc/] connects JavaScript to the message broker
- Multilingual project page (Jekyll / GitHub pages)
- Multilingual manual (livingfire-docbook)

1.0.0 - Overhaul for AngularJS, Neo4j Rest

Version 1.0.0 was never released or deployed! It had a working Java backend but Google decided to brake the AngularJS API with version 2 which in a nut shell meant to

reprogram the frontend sooner or later. After some research 1.0.0 was "declared dead" beacause Spring not only showed how to to solve the frontend problem elegantly with React.js and Spring Data REST [https://spring.io/guides/tutorials/react-and-spring-data-rest/] but it also introduced Level 3 - Hypermedia Control [https://martinfowler.com/articles/richardsonMaturityModel.html] into Skylars REST API.

Di 7. Jul 12:54:13 CEST 2015

- upgrade to Spring Boot 1.3.0.RELEASE
- AngularJS single pages in skylar-gui
- More Node.js [https://nodejs.org] tools for frontend development: yo, grunt, karma, jasmine, bower, ...
- Single NoSql Database with Neo4j (less dependencies)
- Neo4j Browser + Neo4j Remote Shell added
- Docker added

0.0.8 - Introduced Bower

Mo 6. Jul 22:24:10 CEST 2015

- Bower now handles JavaScript libraries
- Smaller files with .min.js or .min.css

0.0.7 - neo4j note handling added

Fr 15. Mai 18:00:44 CEST 2015

- job interview
- phone interview
- rejection letter
- missing communication
- private memo

0.0.6 - Apache Camel Emailprocessor revised

Do 14. Mai 23:30:03 CEST 2015

- Emailprocessor now works with self signed certificates

0.0.5 - Bugfix Velocity + webradio

Do 30. Apr 14:12:27 CEST 2015

- webradio without cronjob
- Bugfix Velocity template lookup

0.0.4 - Apache Camel 2.15.1

Di 21. Apr 20:32:47 CEST 2015

- corrected reading google api file

0.0.3 - Bugfix Logging

So 19. Apr 16:32:47 CEST 2015

- Log4j -> SLF4J

0.0.2 - Job Application

- added Notes to neo4j
- New interface for job applications
- Neo4j REST connection + auth
- Searchable Bootstrap tables
- New Ports 61666-61669 for skylar
- Log4j dayly rotate activated

0.0.1-hotfix+webradio

Do 5. Mär 22:00:00 CET 2015

MOC - music on console won't work on Windows - replaced with jPlayer

0.0.1-hotfix+touch.display

Do 5. Mär 00:02:46 CET 2015

alpha version - web interface for touch display support - 1080p 1920x1080

0.0.1 - initial version

Di 9. Dez 07:04:53 CET 2014

Pre-Alpha-Version with Spring Boot according to Git log.

Concepts and prehistoric versions

Before Spring Boot [https://projects.spring.io/spring-boot/] and Apache Camel [http://camel.apache.org/components.html] the program existed for quite some time as a JEE server [http://tomee.apache.org/]. In the birthyear 2010 Skylar was equipped with "her

own Computer" that executed a Java command line application [https://www.oracle.com/java/] 24/7. Back then the algorithm was freshly migrated from a multitude of programs written in BASH [https://www.gnu.org/software/bash], Java [https://www.oracle.com/java/], PHP-cli [www.php.net], Perl [https://www.perl.org/] and -->> Brainfuck [https://en.wikipedia.org/wiki/Brainfuck] <<--. How far the algorithms date back is unknown but I started serious computer programming and Linux scripting with my profession as a Java Software Engineer back in 2000.