Developing skills for Amazon Echo

Mainz, 26.01.2017
Moritz Kammerer

About the author

- 2016 Supervised bachelor thesis about NUIs
- 2016 Developed customer project with Amazon Echo

- GitHub: @phxql / Twitter: @phxql
- Blog: https://www.mkammerer.de/blog

- Sourcecode: https://github.com/qaware/iot-hessen-amazon-echo

What is a Amazon Echo?

- Digital assistent from Amazon
- Provides an audio interface
- https://www.youtube.com/watch?v=KkOCeAtKHIc

What’s the cool thing?

- Alexa has built-in features: weather, facts, news, music, …
- And it is also extensible via so called skills

- In this talk: We develop a skill for warehouse management

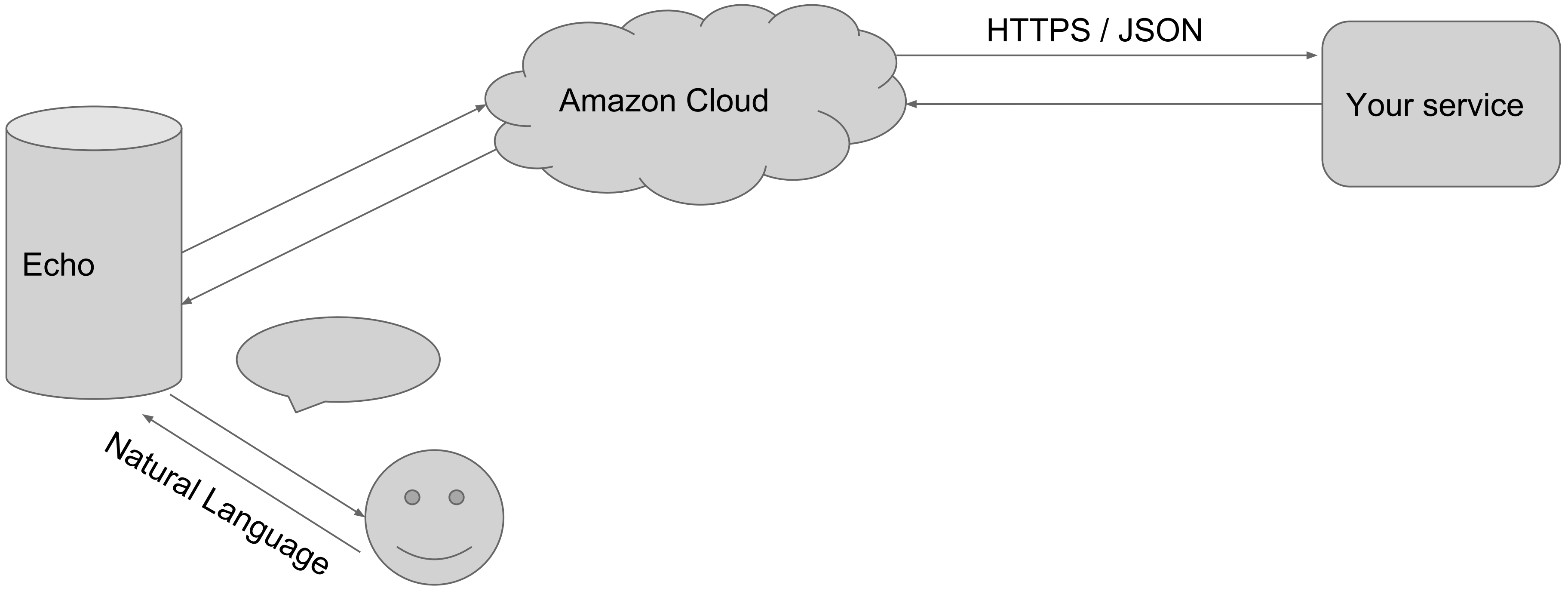
Demo

- Utterances:
  - “Wie viele Schrauben haben wir noch?”
  - “Bestelle mir neue Schrauben”
  - “In welchem Regal befinden sich die Schrauben?”

How does that work?

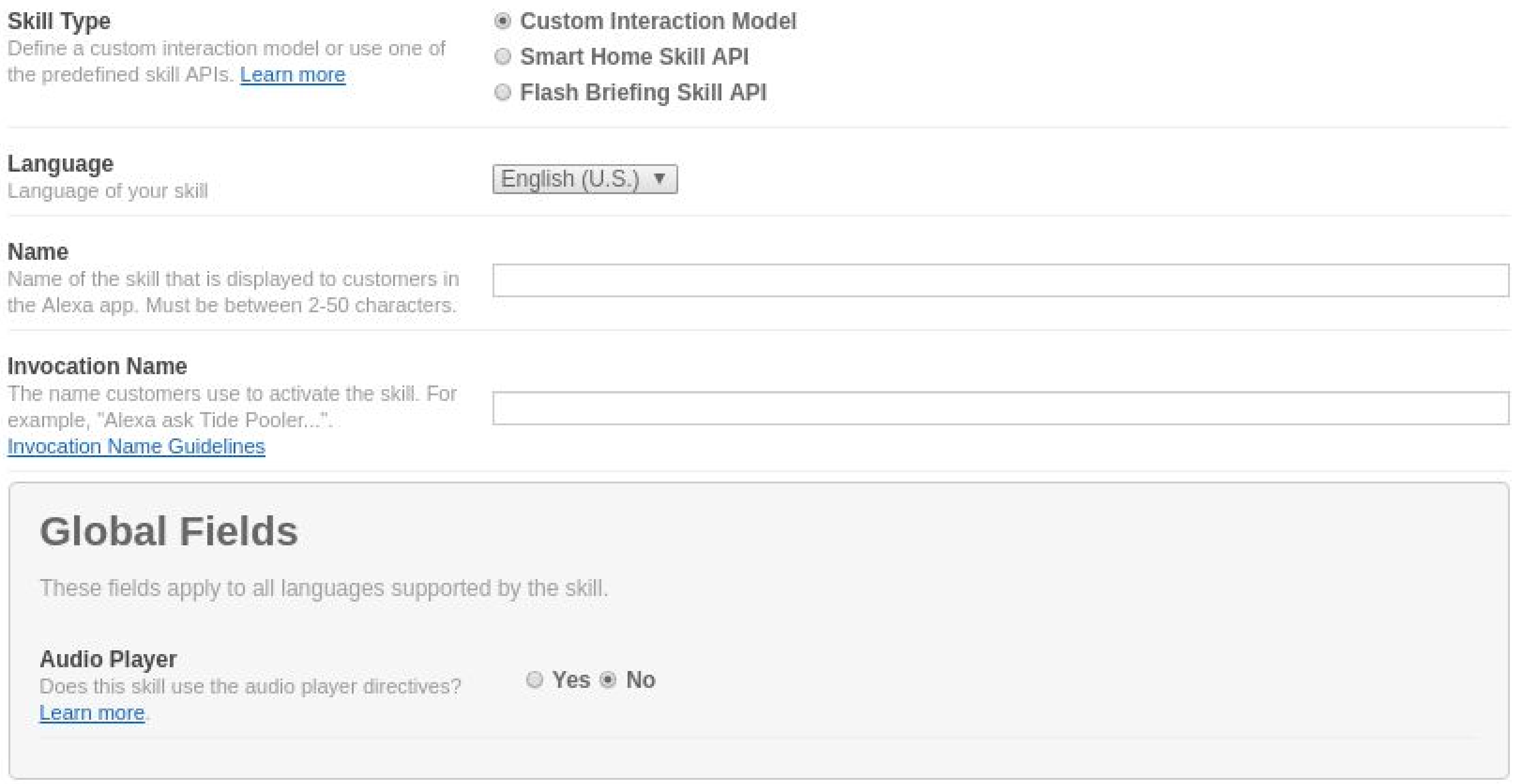
- Register the skill in the Amazon Skill Store
- Develop an application with a HTTP interface
- Enable the skill on your Echo

Okay, but how does that really work?



Develop a skill, step by step. Step 1.

- Create a new skill: Amazon Developer Console / Alexa / Alexa Skill Kit [7]



Skill Type
Define a custom interaction model or use one of the predefined skill APIs. Learn more

[✓ Custom Interaction Model]
Smart Home Skill API
Flash Briefing Skill API

Language
Language of your skill

English (U.S.) ▼

Name
Name of the skill that is displayed to customers in the Alexa app. Must be between 2-50 characters.

Invocation Name
The name customers use to activate the skill. For example, "Alexa ask Tide Pooler...".
Invocation Name Guidelines

Global Fields

These fields apply to all languages supported by the skill.

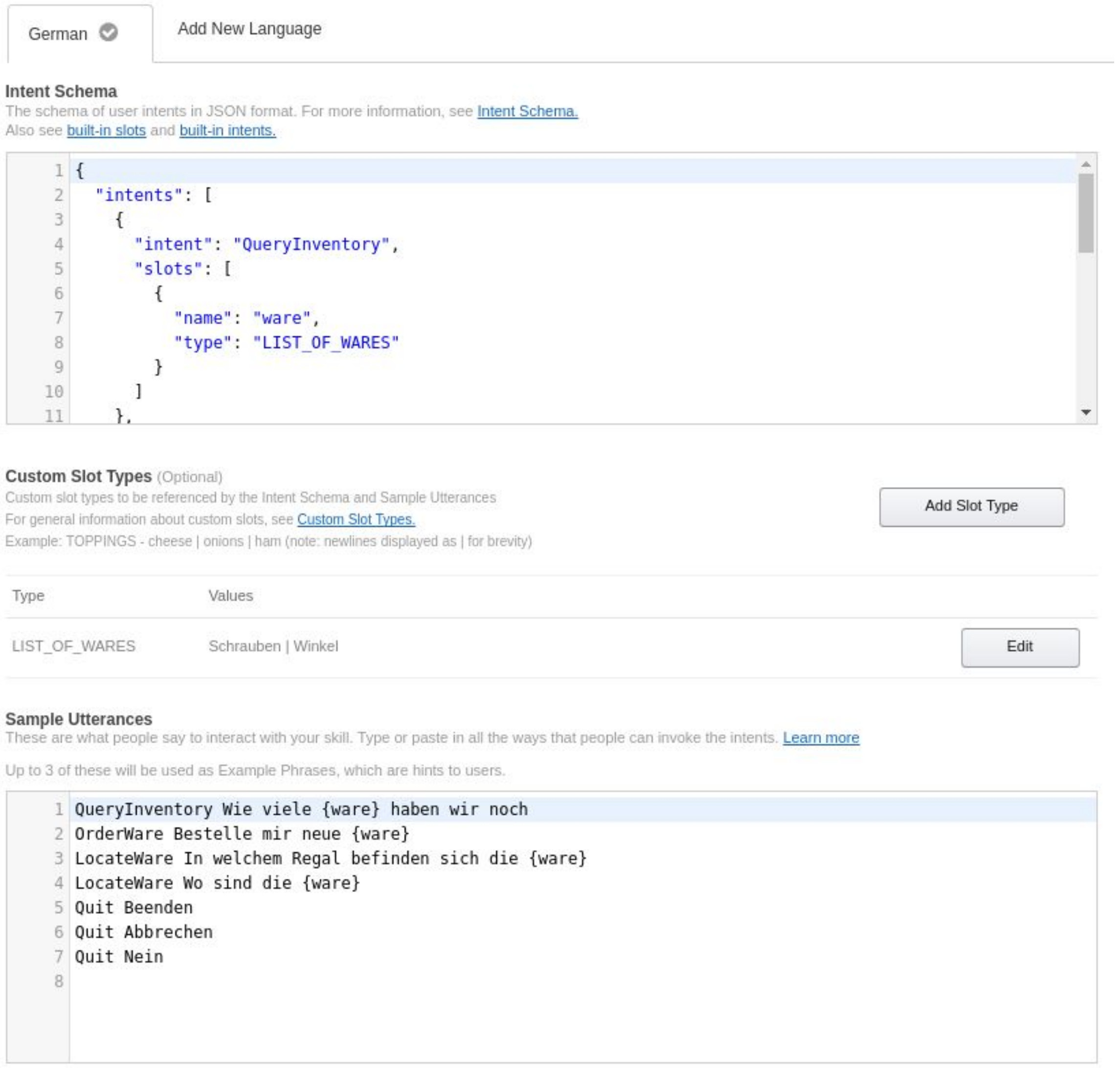
Audio Player
Does this skill use the audio player directives? Learn more.

Yes [✓ No]

Skill types

- SmartHome Skill API: Switch on/off lamps, control heating
- Flash Briefing Skill API: News as audio or text feed
- Custom Interaction Model: Define own utterances, most flexible
    - That’s what we will use

Develop a skill, step by step. Step 2.



Intent Schema
The schema of user intents in JSON format. For more information, see Intent Schema.
Also see built-in slots and built-in intents.

```
{
  "intents": [
    {
      "intent": "QueryInventory",
      "slots": [
        {
          "name": "ware",
          "type": "LIST_OF_WARES"
        }
      ]
    },
```

Custom Slot Types (Optional)
Custom slot types to be referenced by the Intent Schema and Sample Utterances
For general information about custom slots, see Custom Slot Types.
Example: TOPPINGS - cheese | onions | ham (note: newlines displayed as | for brevity)

Type	Values	
LIST_OF_WARES	Schrauben | Winkel	Edit

Sample Utterances
These are what people say to interact with your skill. Type or paste in all the ways that people can invoke the intents. Learn more

Up to 3 of these will be used as Example Phrases, which are hints to users.

```
QueryInventory Wie viele {ware} haben wir noch
OrderWare Bestelle mir neue {ware}
LocateWare In welchem Regal befinden sich die {ware}
LocateWare Wo sind die {ware}
Quit Beenden
Quit Abbrechen
Quit Nein
```

What are intents?

- "an intent represents an action that fulfills a user's spoken request"
- Intent schema is a JSON formatted list of intents

```
{
  "intent": "QueryInventory",
  "slots": [
    {
      "name": "ware",
      "type": "LIST_OF_WARES"
    }
  ]
}
```

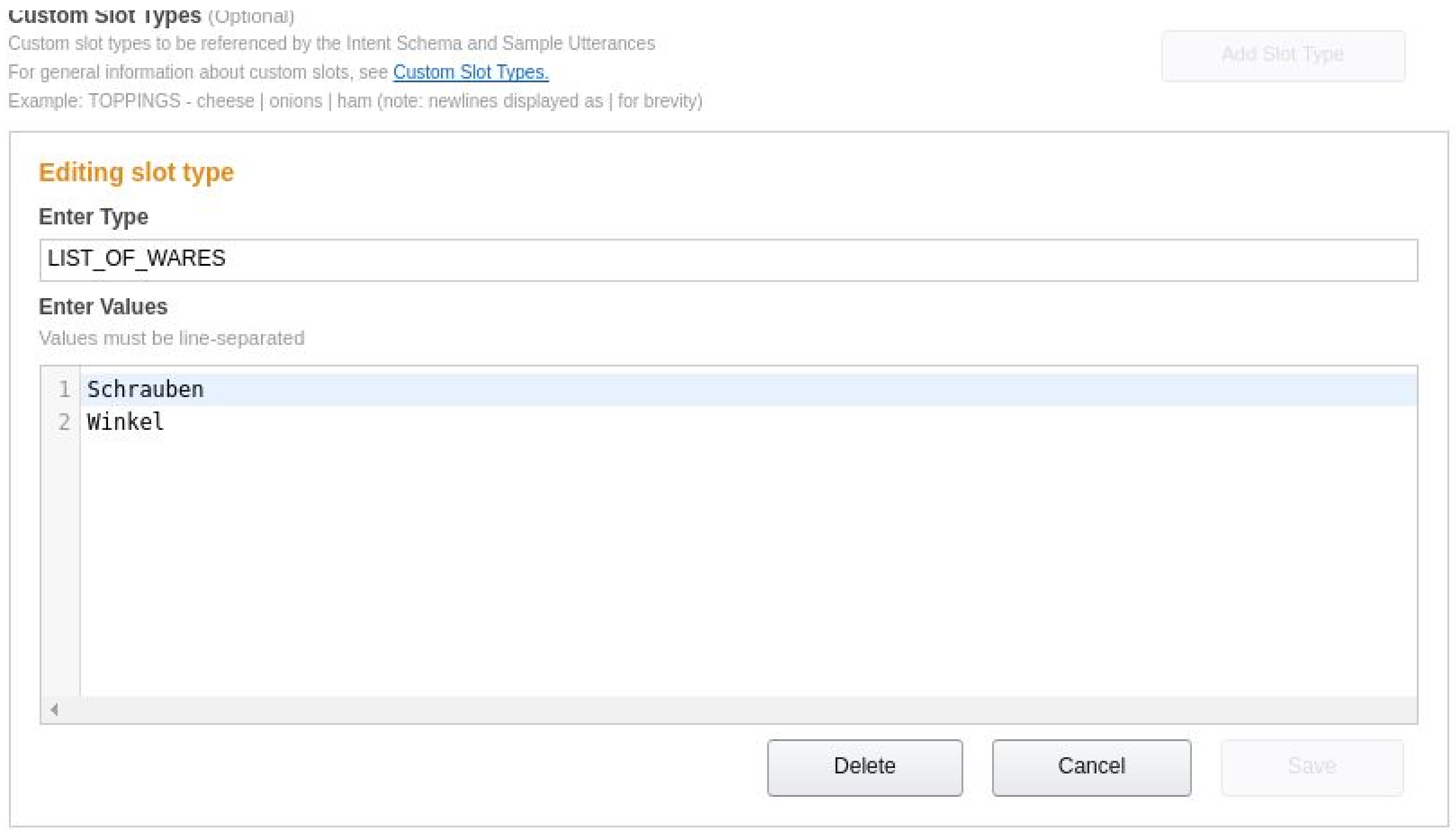
Slots (parameter) of the intent

Slot type

Intents of our skill

- QueryInventory (ware): Determine how many of a ware is in the warehouse
- OrderWare (ware): Orders a new ware
- LocateWare (ware): Find a ware in the warehouse
- Quit (): Aborts the conversation

Intent slot types



Custom Slot Types (Optional)

Custom slot types to be referenced by the Intent Schema and Sample Utterances

For general information about custom slots, see Custom Slot Types.

Example: TOPPINGS - cheese | onions | ham (note: newlines displayed as | for brevity)

Editing slot type

Enter Type

LIST_OF_WARES

Enter Values

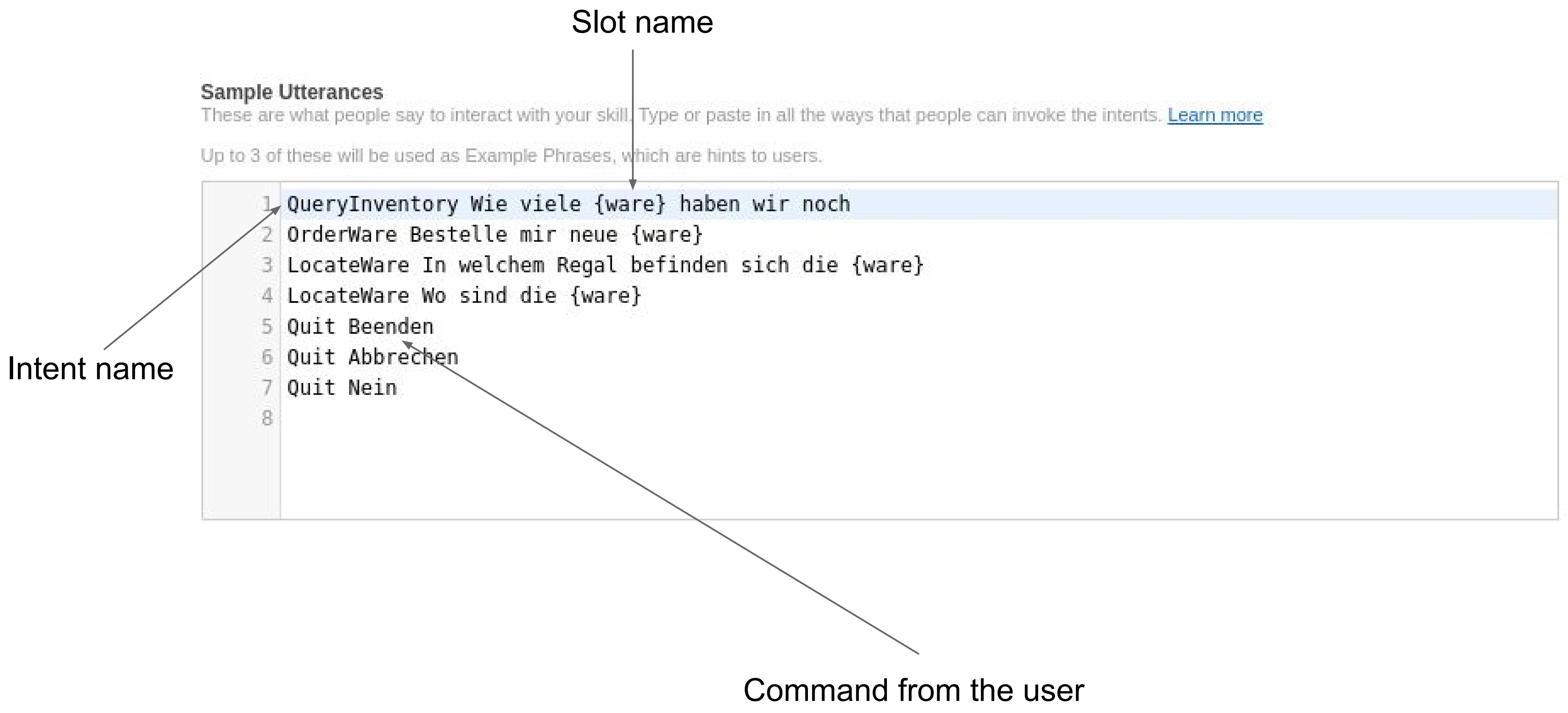
Values must be line-separated

```
Schrauben
Winkel
```

Hint: There are predefined slots, e.g. for dates, durations, time, etc.: [1]

Attention: Alexa tries to match the spoken word to this list, but other values can still be sent to the skill!

Utterances - Combine intents with audio commands



Sample Utterances
These are what people say to interact with your skill. Type or paste in all the ways that people can invoke the intents. Learn more

Up to 3 of these will be used as Example Phrases, which are hints to users.

```
QueryInventory Wie viele {ware} haben wir noch
OrderWare Bestelle mir neue {ware}
LocateWare In welchem Regal befinden sich die {ware}
LocateWare Wo sind die {ware}
Quit Beenden
Quit Abbrechen
Quit Nein
```

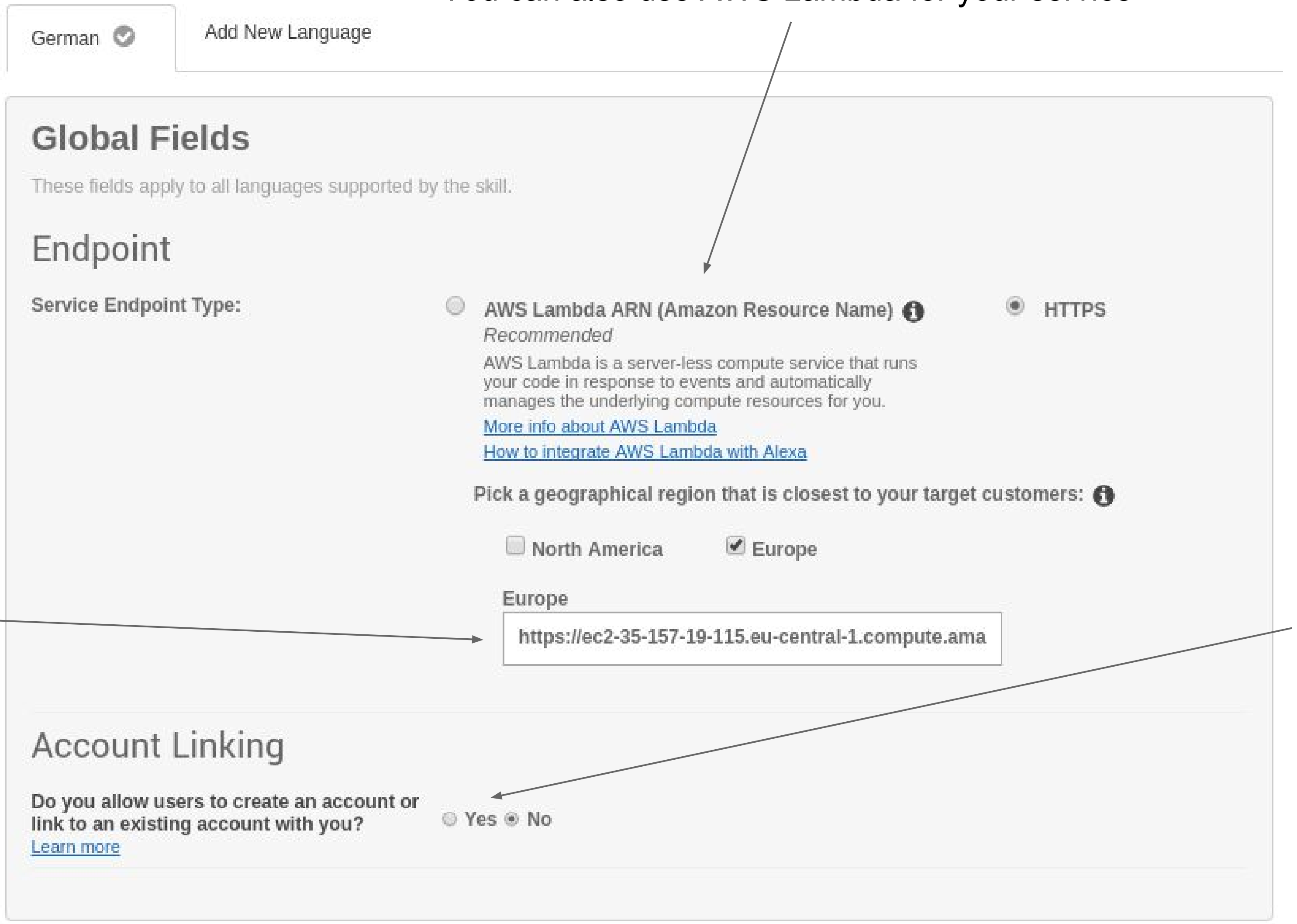
Hint: Best practices and a handbook for the utterances design: [2], [3]

Skill configuration

You can also use AWS Lambda for your service

HTTPS endpoint of our skill

OAuth2 link to an non-Amazon account



German | Add New Language

Global Fields

These fields apply to all languages supported by the skill.

Endpoint

Service Endpoint Type: AWS Lambda ARN (Amazon Resource Name) [✓ HTTPS]

Recommended

AWS Lambda is a server-less compute service that runs your code in response to events and automatically manages the underlying compute resources for you.

More info about AWS Lambda

How to integrate AWS Lambda with Alexa

Pick a geographical region that is closest to your target customers:

North America [✓ Europe]

Europe

https://ec2-35-157-19-115.eu-central-1.compute.ama

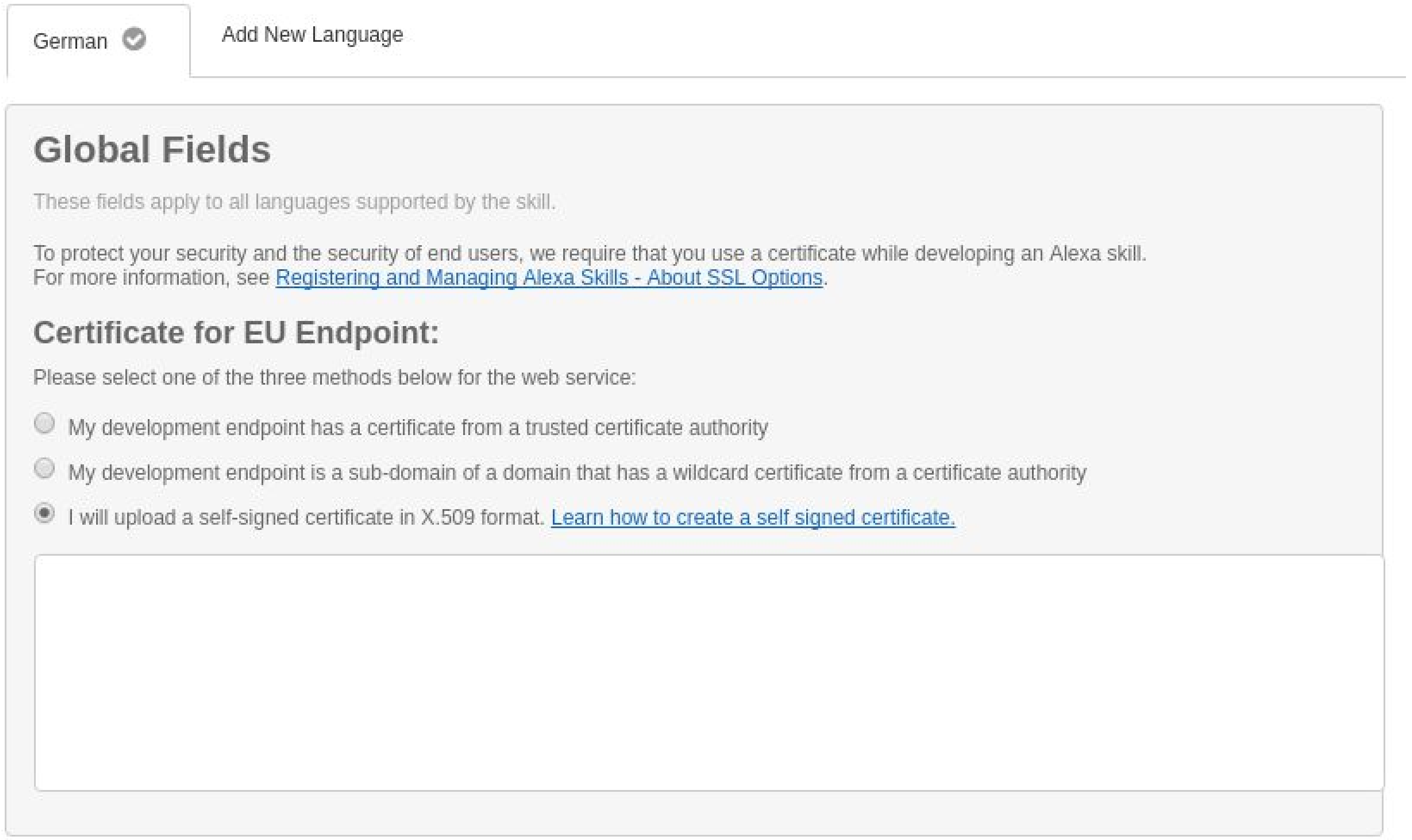
Account Linking

Do you allow users to create an account or link to an existing account with you?

Learn more

Yes [✓ No]

SSL configuration


German ✓ | Add New Language

Global Fields

These fields apply to all languages supported by the skill.

To protect your security and the security of end users, we require that you use a certificate while developing an Alexa skill.
For more information, see Registering and Managing Alexa Skills - About SSL Options.

Certificate for EU Endpoint:

Please select one of the three methods below for the web service:

- ○ My development endpoint has a certificate from a trusted certificate authority
- ○ My development endpoint is a sub-domain of a domain that has a wildcard certificate from a certificate authority
- ● I will upload a self-signed certificate in X.509 format. Learn how to create a self signed certificate.

Skill configured, time for some code!

- We create a Spring Boot application for our service
- Open Spring Initializr [6], dependencies: only web
- Add the Alexa Skills Kit for Java [4] to the Maven POM:

```
<dependency>
    <groupId>com.amazon.alexa</groupId>
    <artifactId>alexa-skills-kit</artifactId>
    <version>1.2</version>
</dependency>
```

Implement the Speechlet

`com.amazon.speech.speechlet.SpeechletV2`

- `void onSessionStarted(...)`
  - Gets called when a session is started
- `SpeechletResponse onLaunch(...)`
  - Gets called when the user starts a conversation
- `SpeechletResponse onIntent(...)`
  - Gets called when the user invokes an intent
- `void onSessionEnded(...)`
  - Gets called when the session is ended

Our speechlet implementation

- onSessionStarted: not needed
- onLaunch:
  - When called, user wants a conversation. Set a session flag:

```
requestEnvelope.getSession().setAttribute("conversation", "true");
```

  - When not called, user wants a one-shot intent
- onSessionEnded: not needed

The skill logic resides in onIntent

- onIntent: read the intent name and handle the intent

```
Intent intent = requestEnvelope.getRequest().getIntent();
switch (intent.getName()) {
    case "QueryInventory":
        return handleQueryInventory(requestEnvelope);
    …
}
```

QueryInventory intent handling

- Read slot “ware”:

```
Slot wareSlot = intent.getSlot("ware");
String ware = wareSlot == null ? null : wareSlot.getValue();
```

QueryInventory intent handling

SSML [5] is also supported

- If ware is missing from the intent `(ware == null)`, tell the user.

- If in conversation mode, let the user retry:
  - `return SpeechletResponse.newAskResponse(new PlainTextOutputSpeech("Ich habe die Ware nicht verstanden. Was möchten Sie tun?"),`

    `new Reprompt(...));`

If the user doesn’t answer the quesion, this text is spoken.

- If not in conversation mode (one-shot intent)
  - `return SpeechletResponse.newTellResponse(new PlainTextOutputSpeech("Ich habe die Ware nicht verstanden."));`

QueryInventory intent handling

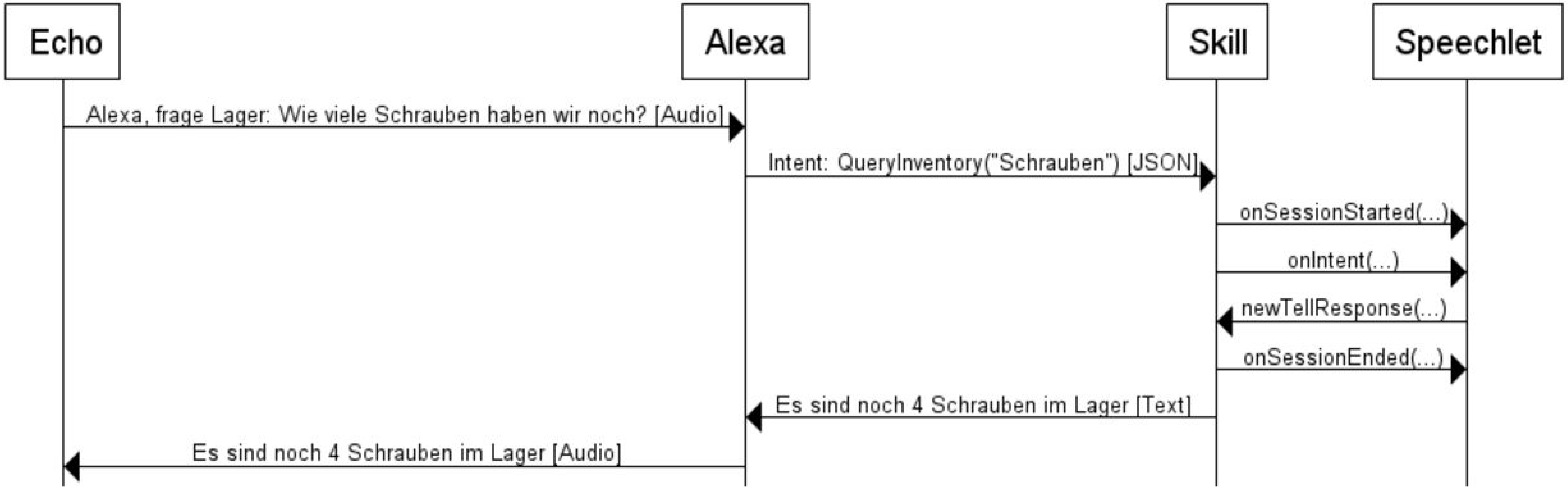
- Now find the ware amount:

```
int amount = warehouseService.getAmount(ware.get());
```

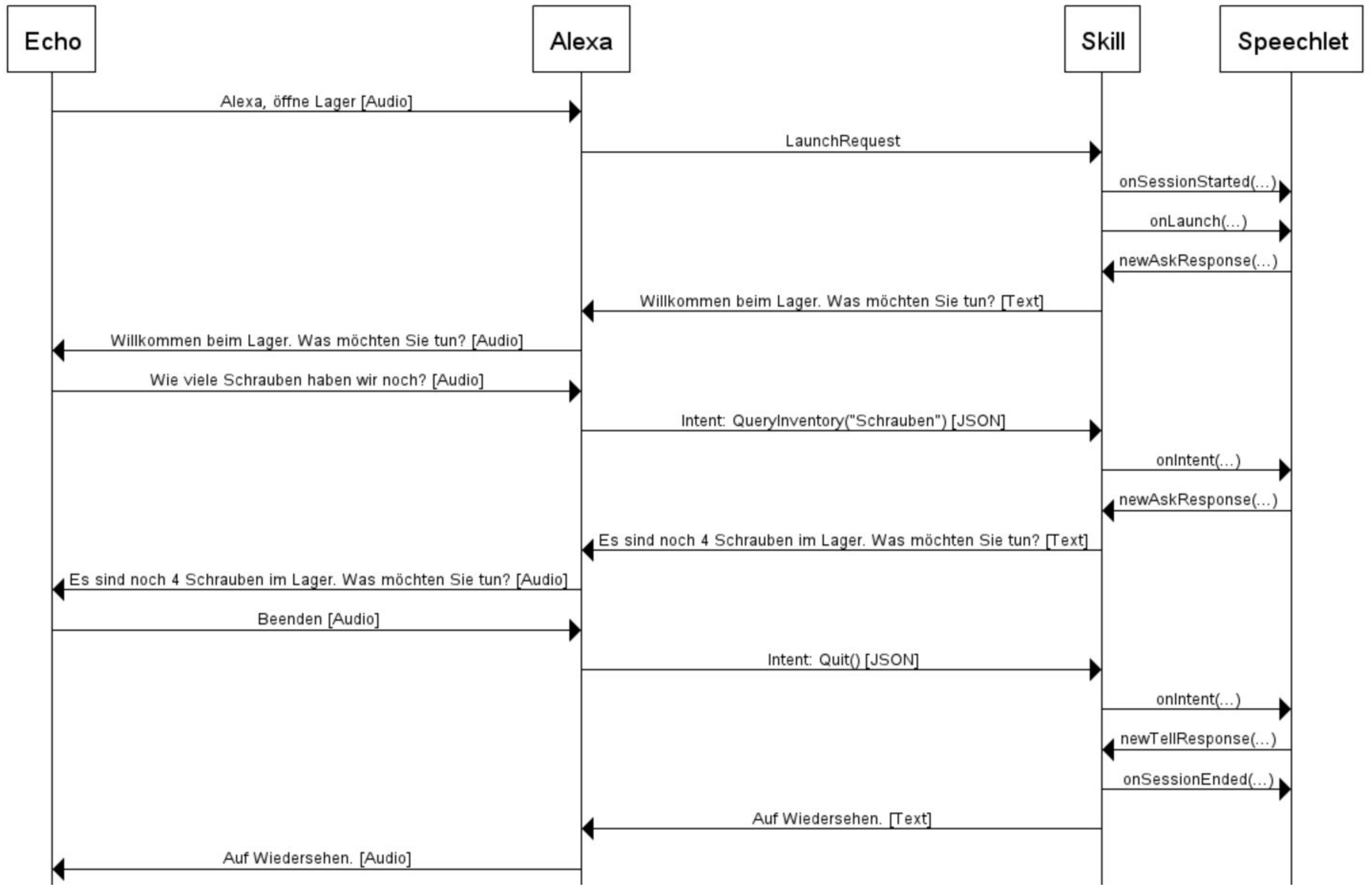
- And create a response to the user:

```
return SpeechletResponse.newTellResponse(new PlainTextOutputSpeech(
   String.format("Es sind noch %d %s im Lager.", amount, ware)
));
```

One-shot flow


Echo
Alexa
Skill
Speechlet
Alexa, frage Lager: Wie viele Schrauben haben wir noch? [Audio]
Intent: QueryInventory("Schrauben") [JSON]
onSessionStarted(...)
onIntent(...)
newTellResponse(...)
onSessionEnded(...)
Es sind noch 4 Schrauben im Lager [Text]
Es sind noch 4 Schrauben im Lager [Audio]

Conversation flow



Echo
Alexa
Skill
Speechlet
Alexa, öffne Lager [Audio]
LaunchRequest
onSessionStarted(...)
onLaunch(...)
newAskResponse(...)
Willkommen beim Lager. Was möchten Sie tun? [Text]
Willkommen beim Lager. Was möchten Sie tun? [Audio]
Wie viele Schrauben haben wir noch? [Audio]
Intent: QueryInventory("Schrauben") [JSON]
onIntent(...)
newAskResponse(...)
Es sind noch 4 Schrauben im Lager. Was möchten Sie tun? [Text]
Es sind noch 4 Schrauben im Lager. Was möchten Sie tun? [Audio]
Beenden [Audio]
Intent: Quit() [JSON]
onIntent(...)
newTellResponse(...)
onSessionEnded(...)
Auf Wiedersehen. [Text]
Auf Wiedersehen. [Audio]

Wire the speechlet into Spring Boot

Speechlet will answer on /alexa

```
@Bean
public ServletRegistrationBean alexaServlet(WarehouseSpeechlet speechlet) {
  SpeechletServlet speechServlet = new SpeechletServlet();
  speechServlet.setSpeechlet(speechlet);

  ServletRegistrationBean servlet = new ServletRegistrationBean(speechServlet, "/alexa");
  servlet.setName("alexa");

  return servlet;
}
```

Set speechlet properties

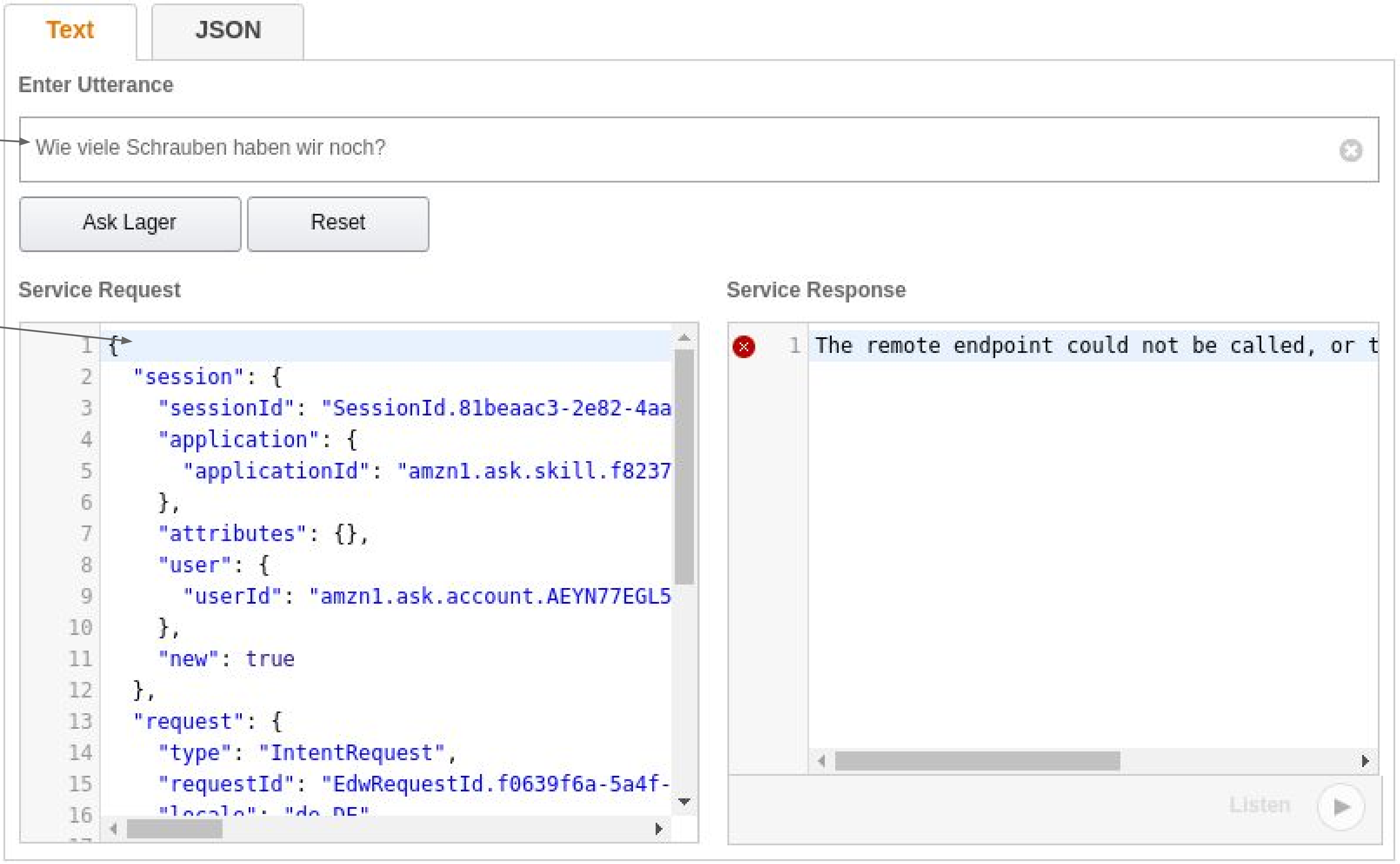
```
// Disable signature checks for development
System.setProperty(Sdk.DISABLE_REQUEST_SIGNATURE_CHECK_SYSTEM_PROPERTY, "true");

// Allow all application ids for development
System.setProperty(Sdk.SUPPORTED_APPLICATION_IDS_SYSTEM_PROPERTY, "");

// Disable timestamp verification for development
System.setProperty(Sdk.TIMESTAMP_TOLERANCE_SYSTEM_PROPERTY, "");
```

For production you’ll find this ID in the “Skill Information” section

Skill implemented, how to test it?

Enter text here

This request gets sent to the skill service



Wie viele Schrauben haben wir noch?

```
{
  "session": {
    "sessionId": "SessionId.81beaac3-2e82-4aa
    "application": {
      "applicationId": "amzn1.ask.skill.f8237
    },
    "attributes": {},
    "user": {
      "userId": "amzn1.ask.account.AEYN77EGL5
    },
    "new": true
  },
  "request": {
    "type": "IntentRequest",
    "requestId": "EdwRequestId.f0639f6a-5a4f-
    "locale": "de-DE"
```

```
The remote endpoint could not be called, or t
```

Hint: Copy the JSON and POST it with your preferred tool (curl, Postman, etc.) to /alexa

And now with voice...

- Amazon forces you to use SSL
- So... configure Spring Boot to use SSL
  - Create a new keystore
  - Create a new keypair, the CN must be set to your servers DNS name
  - Enable SSL in Spring Boot:

```
server.port: 443
server.ssl.key-store: classpath:keystore.jks
server.ssl.key-store-password: ""
server.ssl.key-store-type: jks
server.ssl.key-alias: ec2-35-157-19-115.eu-central-1.compute.amazonaws.com
```

And now with voice...

- Export the SSL certificate as X.509 (starts with -----BEGIN CERTIFICATE-----)
- Paste the X.509 certificate in the skill configuration under “SSL Certificate”
- Build the Spring boot application
- Upload the application to a publicly accessible server (e.g. EC2)
- Start the application
- Add your Alexa to the account which created the skill
- Enable the skill in the Alexa App
- Now you can invoke the skill

What I wish I knew

- TLS: **Must** be port 443, the CN in the certificate **must** match the DNS
- Slot types are not enums, but only recommendations for the speech recognition
- Slots can be null! (e.g. “Bestelle mir neue “)
- The user id is changing when the user removes and re-adds the skill
- Alexa Skill Kit for Java: Exclude `log4j` and `slf4j-log4j12`
- When testing the skill, use the Alexa Mobile App: The cards contain useful debug information
- Implement local and test with voice: Use SSH remote port forwarding:

```
ssh -R 443:localhost:8443 root@server
```

What have we learned?

- Develop an interaction model:
  - Intents, slots and slot types
  - Utterances
- To use the Alexa Skills Kit and implement the SpeechletV2 interface
  - When onSessionStarted(), onLaunch(), onIntent() and onSessionEnded() are called
  - What the difference between newAskResponse() and newTellResponse() is
- How to beat TLS

Conclusion

- Pro:
    - Echo can be extended with custom skills
    - Skills are very flexible
    - The Alexa Skills Kit abstracts all the request and response handling
    - Amazon handles the speech-to-text and text-to-speech
- Contra:
    - No semantic analysis, just matching the utterances exactly
    - Have to invoke the skill by saying, “Alexa, tell [skill] …”
    - TLS is very annoying (and badly documented) when developing a skill

Source Code:

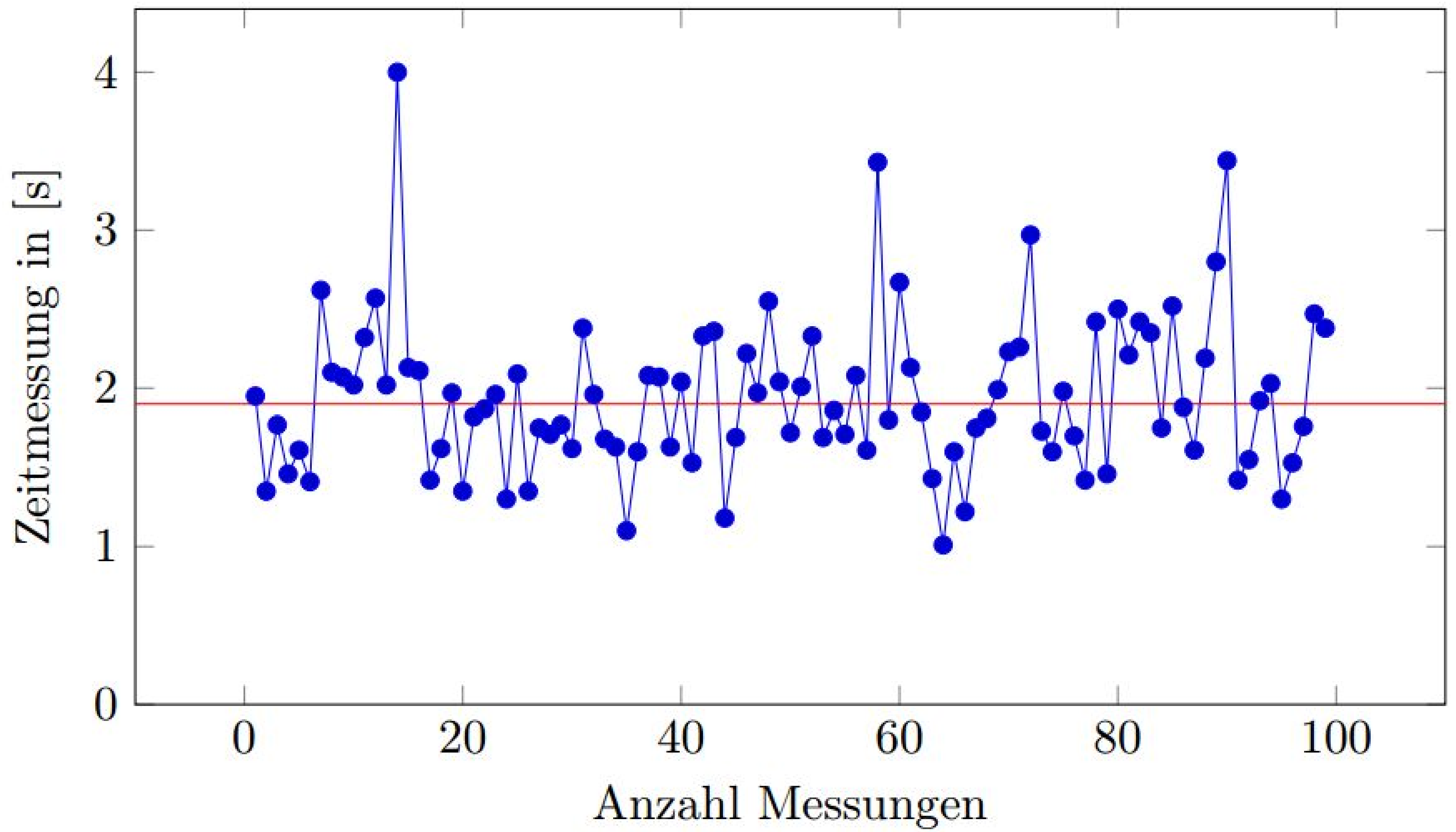
https://github.com/qaware/iot-hessen-amazon-echo

References

- [1] https://developer.amazon.com/public/solutions/alexa/alexa-skills-kit/docs/built-in-intent-ref/slot-type-reference
- [2] https://developer.amazon.com/public/solutions/alexa/alexa-skills-kit/docs/alexa-skills-kit-voice-design-best-practices
- [3] https://developer.amazon.com/public/solutions/alexa/alexa-skills-kit/docs/alexa-skills-kit-voice-design-handbook
- [4] https://github.com/amzn/alexa-skills-kit-java
- [5] https://developer.amazon.com/public/solutions/alexa/alexa-skills-kit/docs/speech-synthesis-markup-language-ssml-reference
- [6] http://start.spring.io/
- [7] https://developer.amazon.com/edw/home.html#/skills/list

Appendix

Timings


Zeitmessung in [s]
Anzahl Messungen