OPTIONEN ZUM BEEINFLUSSEN DER AUDIO-AUSGABE VON ECHO

Audio-Wiedergabe

Neben der Sprachausgabe über Text-to-Voice kann ein Alexa Skill auch Audio abspielen.

Die Text-to-Voice-Engine von Amazon ist sehr mächtig und setzt mit der Sprachausgabe von einfachem Text von Anfang an den Maßstab für Sprach-Assistenten. Wer zum Beispiel die aktuell deutliche schwächere TTV Engine von IBMs Watson ausprobiert, merkt, dass die Kunden in Ihrer Erwartungshaltung ein schlechteres Niveau nicht mehr tolerieren

Trotzdem kann es vorkommen, dass einige Wörter oder Phrasen nicht optimal ausgesprochen werden. Abhilfe schafft hier die Speech Synthesis Markup Language (SSML). Um diese verwenden zu können, muss im Response JSON der Type von *PlainText* auf *SSML* geändert und der auszugebende Text im Element *ssml* mit einführendem und abschließenden *speak*-Tag über-geben werden:

```
"outputSpeech":
{
  "type" : "SSML",
  "ssml" : "<speak>Dies ist SSML.
  </speak>"
}
```

Foto: xxxxxxxxxx

Dieses grundlegende Beispiel führt mit dem *speak*-Tag bereits das erste, von den zur Zeit zwölf unterstützten, SSML-Tags ein. Es dient als einleitende und abschließende Klammer, vergleichbar mit der in Webseiten verwendeten HTML-Notation.

Aussprache

Je nach Sprache hat Amazon bereits feste Phrasen hin-terlegte (für Deutsch zur Zeit 181), welche mit dem Tag *say-as* wie folgt aufgerufen werden können:

```
<speak>
    Ein Beispiel für eine Phrase
    <say-as interpret-as='interjection'>Aber hallo.
    </say-as>
</speak>
```

Für diese Phrasen hat Amazon bereits eine vordefinierte Aussprache hinterlegt. Das *say-as*-Tag unterstützt zusätzlich aber auch noch 11 weitere Interpretations-Typen, mit denen definiert wird, welche Art von Information vorliegt und dann eine entsprechend angepasste Ausgabe gewählt werden kann.

Ähnlich funktioniert auch das *w*-Tag, wobei hier mit dem Parameter *role* die Bedeutung des Wortes spezifiziert werden kann. Dadurch ändert sich bei gleicher Schreibweise - insbesondere im Englischen - die Aussprache. Während sich die Rollen *amazon:VB* (Verb im Präsens), *amazon:VBD* (Verb im Präteritum) und *amazon:NN* (Substantiv) noch einfach erschließen, muss man die Rolle *amazon:SENSE_1* (alternative Bedeutung) ausprobieren, da diese nicht weiter dokumentiert ist:

```
<speak>
Obwohl gleich geschrieben, wird <w
role='amazon:VB'>read</w> in der
Vergangenheitsform als
<w role='amazon:VBD'> read</w>
ausgesprochen.
</speak>
```

Völlige Freiheit zum Definieren der Aussprache bietet das *phoneme*-Tag. Dieses enthält als Parameter das zu verwendende phonetische System, das Internationale Phonetische Alphabet (IPA) oder das Extended Speech Assessment Methods Phonetic Alphabet (X-SAMPA) - und die phonetische Aussprache in der jeweiligen Notation. Auf diese Weise ist es möglich, Alexa mit Dialekten sprechen zu lassen, wobei der Text zwischen öffnendem und schließendem Tag ignoriert wird. Dieses Verhalten entspricht auch dem des *sub*-Tag, wobei hier im Parameter *alias* der ausgesprochene Alternativ-Text in Form normalen Texts angegeben wird. Der eingeschlossene Text, welcher nicht ausgesprochen wird, kann aber zum Beispiel für Anzeigen genutzt werden:

```
<speak>
Im Chemielabor ist kein <sub alias='Magnesium'>Mg</sub>
mehr vorhanden.
</speak>
```

Neben der Definition, was ausgesprochen werden soll, bietet SSML auch die Möglichkeit zu definieren, wie et-was ausge-

sprochen wird. Auch hierfür existieren vordefinierte Verfahren. So verändert das *emphasis*-Tag die Lautstärke und Geschwindigkeit, mit der die Textwiedergabe erfolgt - entsprechend drei Abstufungen:

```
<speak>
Noch normal, <emphasis level='moderate'>Etwas lauter und
langsamer - Standard, wenn kein level angegeben ist.
</emphasis>, <emphasis level='strong'>Noch lauter und
langsamer, asl moderate.</emphasis> und <emphasis
le-vel='reduced'>Leiser und schneller als normal.
</emphasis>
</speak>
```

Hinzu kommt noch das proprietäre *amazon:effect*-Tag, mit welchem zurzeit nur ein Effekt möglich ist - das Flüstern:

```
<speak>
<amazon:effect name='whispered'>Das ist geheim!
</<amazon:effect>
</speak>
```

Falls diese vorgegeben Manipulationsmöglichkeiten nicht ausreichen, gibt es mit dem *prosody*-Tag auch wieder ein Tag, mit dem sehr viele Freiheiten konfigurierbar sind:

```
<speak>
  Meine Aussprache kann
  <prosody volume='x-loud'>man lauter machen</prosody>,
  <prosody rate='x-slow'>oder langsamer</prosody>,
  <prosody pitch='x-high'>oder höher</prosody>,
  <prosody volume='x-loud' rate='x-slow' pitch=
  'x-high'>und kombinieren läßt sich das auch</prosody>.
</speak>
```

Über den Parameter *rate* kann die Geschwindigkeit der Aussprache auf bis zu einem Fünftel (20 Prozent) verringert werden, während 100 Prozent der normalen Geschwindigkeit entspricht und Werte über 100 Prozent die Geschwindigkeit erhöhen. Ein Limit ist hier nicht definiert. Alternativ können auch vordefinierte Werte zum Beispiel *slow* oder *fast* verwendet werden.

Die Tonhöhe der Aussprache lässt sich mit dem Para-meter *pitch* verändern, wobei diese Veränderung von 33 Prozent tiefer bis 50 Prozent höher begrenzt ist. Auch hier existieren vordefinierte Werte wie zum Beispiel *low* oder *high*.

Die Lautstärke lässt sich über den Parameter *volume* verändern, und muss in der Einheit Dezibel angegeben werden. Die vorhergehenden Parameter beziehen sich immer auf die Standard-Aussprache, sodass auch die Prozentangaben dadurch absoluten Werten entsprechen. Im Unterschied dazu bezieht sich die Dezibel-Angabe von *volume* relativ auf die aktuell verwendete Lautstärke. Aber natürlich existieren auch hier vordefinierte absolute Werte wie zum Beispiel *silent* oder *loud*.

All diese Veränderungen lassen sich auch kombinieren, wobei zu beachten ist, dass dies zu einer Verstärkung führen kann beziehungsweise sich einzelne Effekte gegenseitig aufheben. Eine Beschleunigung der Aussprache (rate) führt gefühlt auch zu einem Anheben der Tonhöhe (pitch), sodass ein Absenken dieser anscheinend keinen Effekt hat beziehungsweise ein zusätzliches Anheben deutlich stärker wirkt.

Tabelle 1: Alexa Skills Kit Sound Library

Kategorie	# Sounds
Ambience Sounds	2
Sound Library – Animal Sounds	67
Battle Sounds	10
Cartoon Sounds	3
Foley Sounds	29
Home Sounds	25
Human Sounds	42
Game Show Sounds	23
Impact Sounds	8
Magic Sounds	7
Musical Sounds	34
Nature Sounds	31
Office Sounds	9
SciFi Sounds	71
Transportation Sounds	30

Platzieren von Pausen

Die letzte Steuerungsgröße, um die Aussprache möglichst natürlich erscheinen zu lassen, ist das Platzieren von Pausen. Mit dem *break*-Tag werden diese eingeleitet und über den Parameter *time* in Sekunden oder Millisekunden angeben. Auch hier existiert mit dem Parameter *strength* die Möglichkeit, vordefinierte Längen wie *weak* oder *strong* anzugeben:

```
<speak>
    Jetzt kommt eine Pause<break time='3s'>und dann geht
    es weiter.
</speak>
```

Indirekt kann dieses Verhalten auch durch die Verwendung des *p*-Tag erzielt werden, welches eigentlich zum Gruppieren von Absätzen dient. Durch das Tag wird zwischen den Absätzen eine Pause der Länge *x-strong* eingefügt.

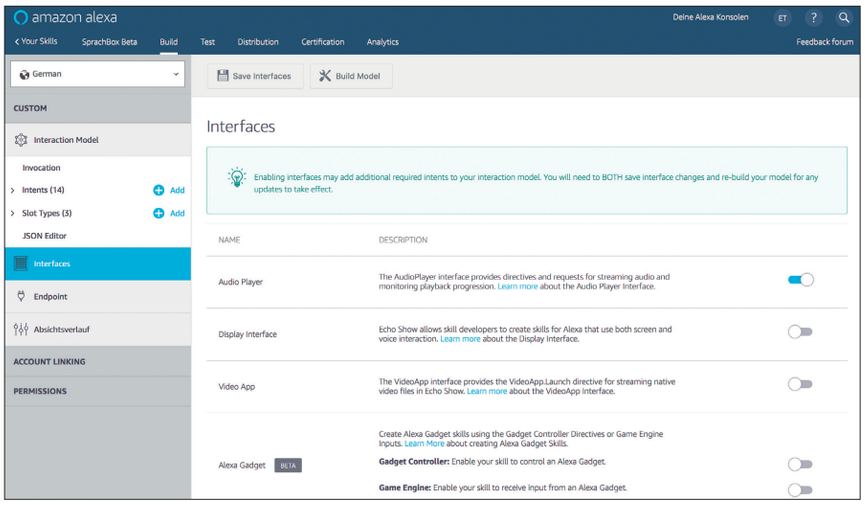
Ähnlich funktioniert das *s*-Tag, welchen einzelne Sätze abgrenzt und einer Pause der Länge *strong* entspricht. Diese Pause hält Alexa alternativ auch nach einem Satzpunkt ein:

```
<speak>
  <p>Die Pause zwischen Paragraphen
```

```
  </p>
  <p>ist länger<s>als die Pause
  </s><s>zwischen Sätzen</s></p>
</speak>
```

Neben der Steuerung der Text-To-Voice Funktionalität, kann SSML auch dazu genutzt werden, reine Audio In-halte zu integrieren. Hierzu dient das audio-Tag, welches den Parameter *src* mit einer URL auf die Sound-Datei enthält.

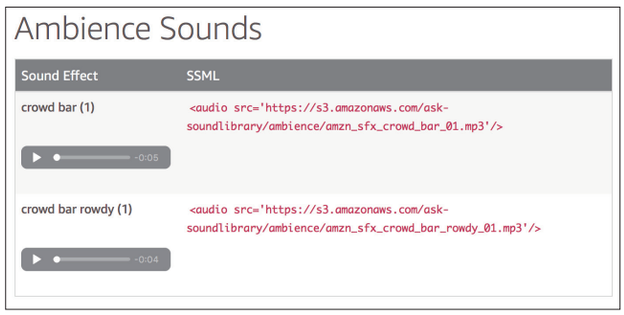
An die referenzierten Sound-Dateien stellt Amazon eine Reihe von Bedingungen. Es muss sich um eine öffentlich erreichbare HTTPS-URL handeln, welche ein trusted SSL Zertifikat nutzt (selbstsignierte Zertifikate sind nicht zugelassen). Und die Datei muss eine valide MP3-Datei (MPEG 2-Format) sein, welche mit einer Bit-Rate von 48 kbit/s und einer Samplingrate von 16 kHz erstellt wurde. Schließlich darf die Länge nicht mehr als 90 Sekunden betragen:



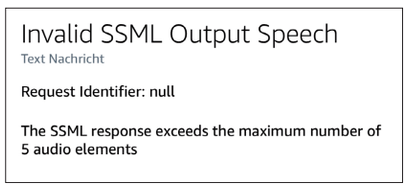
Alexa Skill Konfiguration, hier muss das Audio Player Interface aktiviert werden (Bild 3)

```
<speak>
  So <audio src='https://s3.amazonaws.com/
  ask-soundlibrary/animals/amzn_sfx_lion_roar_01.mp3'/>
  klingt ein Löwe.
</speak>
```

Diese Einschränkungen schließen zwar eine HiFi-Wiedergabe von Musik aus, die Qualität ist jedoch für das Einspielen von Geräuschen und wiedererkennbaren Jingles ausreichend. Um den Einstieg einfach zu halten und damit man sich nicht erst mit dem Umcodieren von Sound-Dateien beschäftigen muss, stellt Amazon unterteilt in 14 Kategorien insgesamt 391 Sounds zur Verfügung (Tabelle 1). In jeder Kategorie werden die Sounds in einer Tabelle aufgelistet, wobei neben dem Namen und Code zum Einbinden des jeweiligen Sounds, auch direkt ein Audioplayer zum Vorhören verfügbar ist (Bild 1). Abschließend sei noch erwähnt, dass Amazon die An-zahl von Audio Tags pro SSML Text auf fünf limitiert, dies aber nicht dokumentiert hat. Bei Verwendung von mehr als fünf SSML Tags erhält man als Sprachausgabe die Information, dass ein Fehler aufgetreten ist. Erst in der App oder auf einem Echo Show beziehungsweise Sport erfährt man den wahren Grund (Bild 2).



Amazon Sound in der Ambience Sound Kategorie der Alexa Skill Kit Sound Library (Bild 1)



Alexa Card Fehlermeldung bei Benutzung von mehr als 5 Sound Tags im SSML Response (Bild 2)

Audio-Player Interface

Wer längere Audio Clips, die darüber hinaus auch noch eine bessere Sound-Qualität haben dürfen, abspielen möchte, dem bleibt nur das Audio Player-Interface. Dieses muss zuerst in der Skill-Konfiguration aktiviert werden (Bild 3). Hiermit werden dem Skill auch automatisch die beiden verpflichtenden Build-In Intents *AMAZON.PauseIntent* und *AMAZON.ResumeIntent* hinzugefügt, welche ein Skill mindestens unterstützen muss. Darüber hinaus gibt es noch neun optionale Build-In Intents, die man unterstützen sollten, selbst wenn man nur eine Rückmeldung geben möchte, dass eine bestimmte Funktionalität - wie zum Beispiel das Mischen der Wiedergabeliste - beim jeweiligen Skill keinen Sinn ergibt.

An diesen Intents erkennt man schon den entscheidenden Unterschied beim Abspielen - es findet außerhalb des eigenen Skills statt. Der Vorteil ist natürlich eine schnellere Steuerung zum Beispiel mit »Alexa, weiter!«. Hierbei wird der Intent *AMAZON.NextIntent* des Skills aufgerufen, welches als letztes mit dem Audio-Player Interface interagiert hatte. Laut Dokumentation sollte das Skill sowohl mit den normalen Antworten, als auch mit Kommandos für den Audio-Player (sogenannten Audio-Player Directives) antworten können. Zwar führt dies nicht zu einem Fehler, aber auch nicht zu dem dokumentierten Verhalten, dass zuerst die normale Ausgabe erfolgt und dann der Audio-Player agiert. Es wird nur die Au-

Listing 1: Alexa Anfrage erhalten

```
// Den Datenstream der Anfrage erhalten
$postPlain = file_get_contents( 'php://input' );

// Decode the JSON into a stdClass object
$alexa_request = json_decode( $postPlain );

if (isset($alexa_request->context->System->
device->supportedInterfaces->AudioPlayer))
    {
       //Wenn Audio Interface unterstützt wird …
    } else     {
       //Wenn KEIN Audio Interface unterstützt wird
…
    }
```

Listing 2: Audio-Player Play Directive

```
{
  "version": "1.0",
  "response": {
"directives": [{
"type": "AudioPlayer.Play",
"playBehavior": "REPLACE_ALL",
"audioItem": {
"stream": {
"token": "MyToken1234",
"url": "https://MyDomain.com/Sound.MP3",
"offsetInMilliseconds": 0
    }}}],
    "shouldEndSession": false
    },
"sessionAttributes": {}
}
```

dio-Player Directive aufgeführt. Ebenso wird auch das *shouldEndSession*-Attribut ignoriert, sodass eine Audio-Player Directive faktisch einem Ausstieg aus dem Skill entspricht.

Bevor ein Audio-Player Interface verwendet wird, sollte das Skill prüfen, ob das genutzte Endgerät diese auch unterstützt (Listing 1), da ansonsten Fehler ausgelöst werden. Zwar unterstützten eigentlich alle Geräte der Echo-Familie das Audio-Player Interface, aber von der Fire-TV-Familie wird es nur vom neusten Endgerät - dem Fire Cube - unterstützt und darüber hinaus steigt momentan auch die Anzahl der Alexa supported Devices, bei denen nicht einfach von einer Unterstützung ausgegangen werden kann.

Wie bereits erwähnt, wird der Audio-Player in der Skill Antwort mit einer Directive angesprochen, welches ein Array ist, für das es zurzeit vier verschiedene Klassen gibt. Dabei darf je Klasse nur maximal ein Objekt enthalten sein.

Zur Steuerung des Audio-Player-Interfaces stehen drei Directives zur Verfügung: *AudioPlayer.Play*, *AudioPlayer.Stop* und *AudioPlayer.ClearQueue*. In Listing 2 ist ein minimale *AudioPlayer.Play*-Directive zu sehen, welche am verwendeten Type zu erkennen ist. Durch das Attribute *playBehavior* wird festgelegt, wie mit der Audio-Information umgegangen wird:

- *REPLACE_ALL*: Die Wiedergabeliste wird gelöscht und die Audio-Information sofort abgespielt, auch wenn bereits eine andere Wiedergabe aktiv war.
- *REPLACE_ENQUEUED*: Die aktuelle Wiedergabe wird weiter abgespielt, die Wiedergabeliste jedoch gelöscht und diese Audio-Information wird der Wiedergabeliste hin- ▶

Listing 3: Audio-Player Request

```
{
  "version":"1.0",
  "context": {
    "AudioPlayer": {
      "offsetInMilliseconds": 0,
      "token":"1234"
    },
   "Display": { "token":"A20793"},
   "System": {
     "application": {"applicationId":"APPLICATIONID"},
     "user": {"userId":"USERID"},
     "device“: {
       "deviceId":"DEVICEID",
       "supportedInterfaces":{
         "AudioPlayer":{},
         "Display":{
           "templateVersion":"1.0",
           "markupVersion":"1.0"
         },
         "VideoApp":{}
     }},
     "apiEndpoint":"https://api.eu.amazonalexa.com",
     "apiAccessToken":"BlaBla"
   }},

   "request":{
     "type":"AudioPlayer.PlaybackStarted",
     "requestId":"REQUESTID",
     "timestamp":"2018-04-24T08:59:16Z",
     "locale":"en-GB",
     "token":"1234",
     "offsetInMilliseconds":0
   }
}
```

zugefügt. Erst nach Abspielen der aktuellen Wiedergabe wird die Audio-Information abgespielt.

- *ENQUEUE*: Die Audio-Information wird der Wiedergabeliste hinzugefügt. Erst nach Abspielen der aktuellen Wiedergabe und aller vorherigen Einträge der Wiedergabeliste wird die Audio-Information abgespielt.

Es folgt ein *audioItem*-Objekt, welches ein *stream*-Objekt enthält, in dem die eigentlichen Attribute zur Audio-Datei enthalten sind. Dies ist zum einen die URL, welche HTTPS mit einem trusted Zertifikat (nicht selbstsigniert) nutzt und auf eine Datei mit dem Format AAC/MP4, MP3 oder HLS verweist. Die Bitrate muss zwischen 16kbps und 384 kbps betragen, was achtmal so viel ist wie bei Audio-Dateien, die per SSML eingebunden werden können.

Zusätzlich wird noch ein Offset in Millisekunden angegeben, um die Wiedergabe auch von einer beliebigen Position aus starten zu können. Dies ist zum Beispiel wichtig, wenn man mit dem Intent *AMAZON.ResumeIntent* eine angehaltene Audio-Datei weiter abspielen und nicht von vorne starten möchte.

Audio-Player Requests

Doch woher sollte das Skill wissen, bei welchem Offset eine Audio-Wiedergabe gestoppt wurde. Hierfür gibt es die sogenannten Audio-Player Requests. Diese agieren ähnlich wie Intent-Aufrufe, allerdings mit zwei Unterschieden (Listing 3). Erstens wird das Session-Objekt im Request nicht mitgeliefert. Zweitens kann in der Response nur eine Audio-Player-Directive stehen und keine Antwort für Alexa zum Ausgeben, zum Beispiel *outputSpeech*. Ein Response ist aber nicht verpflichtend, da es dabei vor allem eine Information zur Änderung des Audio-Player Status handelt (Listing 4).

Diese beginnen mit *AudioPlayer.Playback* und reichen von *Startet* über *Stopped* bis *Finished* für den normalen Lifecycle. Es kann aber auch über *Failed* und *NearlyFinished* informiert werden. Gerade Letzteres ist etwas fehlleitend, da der Request *AudioPlayer.PlaybackNearlyFinished* nicht kurz vor Ende des Abspielens ausgelöst wird, sondern wenn der Stream fertig geladen wurde. Dies kann auch kurz nach dem Starten passieren, auch wenn noch einige Minuten zum Abspielen vorhanden sind. Es dient in erster Linie dazu, die nächste Audio-Datei in die Wiedergabeliste hinzuzufügen.

Listing 4: Pausing Audio-Player

```
session_id($alexa_request->context->System->
user->userId);
session_start();
if  ($alexa_request->request->intent->name ==
"AMAZON.PauseIntent")
{
   $_SESSION['OffSet'] = $alexa_request->
   request->offsetInMilliseconds;
// Speichern des Offsets, um es beim
// AMAZON.ResumeIntent zu verwenden
?>
{
"version": "1.0",
"response": {
"directives": [{
    "type": "AudioPlayer.Stop"
 }]}
}
<?
}
session_commit();
```

Links zum Thema

- Amazon Developer Portal – Alexa Skill Kit Dokumentation: Speech Synthesis Markup Language (SSML) Reference **https://developer.amazon.com/de/docs/custom-skills/speech-synthesis-markup-language-ssml-reference.html**
- Liste der von Amazon unterstützten deutschen SSML Phrasen **https://developer.amazon.com/de/docs/custom-skills/speechcon-reference-interjections-german.html**
- Alexa Skills Kit Sound Library **https://developer.amazon.com/de/docs/custom-skills/ask-soundlibrary.html**

Fazit

Die eigenen Alexa-Skills sind nicht auf simple Antworten mit der immer gleichen Stimme beschränkt. Allerdings sollte man die verschiedenen Möglichkeiten mit Bedacht einsetzten. Einzeln eingestreute Effekte schaffen Atmosphäre, zu viele können jedoch schnell als störend empfunden werden. Bevor man sich mit den Effekten austobt, sollte man daher zuerst die Möglichkeit in Betracht ziehen, möglichst viel Abwechslung durch alternative Antworten und variierenden Satzstellungen in die Konversation einzubringen.

Auch wenn der Audio-Player mit seiner möglichen langen Abspielzeit und hohen Audio-Qualität eine nahliegende Lösung ist, sollte einem bewusst sein, dass neben einen erhöhten Aufwand für dessen Einbindung, vor allem das Verlassen einer durchgängigen Interaktion mit dem Skill der Preis dafür ist. ■

Emil Thies
ist in der IT Abteilung der Deutschen Telekom angestellt und probiert auch nebenberuflich neue Technologien aus, indem er eigene Apps, Skills und Actions programmiert und veröffentlicht.