ALEXA-SKILL MIT BENUTZERVERKNÜPFUNG

Alexa personalisieren

So erweitert man eine Alexa-Anbindung um individuelle Inhalte.

In meinem letzten Beitrag habe ich gezeigt, wie ein PHP-Backend mit kleinen Anpassungen um eine Alexa-Anbindung erweitert werden kann. In diesem Beitrag geht es nun darum, wie eine solche Anbindung personalisiert werden kann. Hierfür existieren verschiedene Umsetzungsmöglichkeiten, bei denen man entsprechend des jeweiligen Usecase, Funktionalität und Umsetzungskomplexität gegeneinander abwägen sollte.

Temporäre Personalisierung

Am einfachsten lässt sich eine Nutzung von Daten über mehrere Aufrufe innerhalb derselben Session realisieren. Dies ermöglicht es der Programmlogik - je nach bisheriger Nutzer-Eingabe - unterschiedlich zu reagieren. Dabei werden User-Anfragen von Amazon als Session betrachtet, solange das Backend in der Antwort ein

```
"shouldEndSession": false
```

erhält und die Session nicht abgebrochen wird. Eine offene Session führt dazu, dass Alexa nach der Ansage der Antwort auf eine weitere Nutzeranfrage wartet. Aus diesem Grund achtet Amazon bei der Freigabe darauf, dass am Ende der Antwort eine Aufforderung zur Interaktion beziehungsweise Frage gestellt wird.

Ein Abbruch der Session wird entweder aktiv durch den Nutzer (zum Beispiel durch das Schlüsselwort *Exit*) oder nach Inaktivität (deren Dauer je Echo Type unterschiedlich definiert ist) eingeleitet. Ein solches Session-Ende wird dem Skill mit einem *SessionEndedRequest* signalisiert, sodass darauf explizit reagiert werden kann.

Analog zur Verwendung von Cookies durch Websites, signalisiert das Backend dem Amazon Alexa Skill, dass es Daten temporär zwischenspeichern und diese bei einer weiteren Anfrage an das Skill-Backend zurück schicken soll. Hierzu muss im Skill-Response eine Key-Value-Liste unter *sessionAttributes* abgelegt werden:

```
{
"version": "1.0",
    "response": {
        "outputSpeech": {
            "type": "PlainText",
            "text": "Hallo Welt. Und nun?"
        },
"shouldEndSession": false
    },
"sessionAttributes": {
      "Planet":"Erde"
    }
}
```

Bild: Amazon

Im nächsten Request werden diese in session unter attributes wieder angeliefert. Der JSON-Request wird in ein Objekt geladen und so kann mit *$alexa_request->session->attributes->Planet* auf eine Session-Variable zugegriffen werden, wobei

Planet durch den jeweiligen Key-Namen ersetzt werden muss:

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

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

Hier erfolgt die Skill-Aktivierung auf der Amazon-Webseite (Bild 1)

Die Session Variablen müssen nicht zwangsläufig an Amazon übertragen werden. Auch das Skill-Backend kann diese zwischenspeichern, muss dann aber seine eigene Session mit der von Amazon verknüpfen. Zum eindeutigen Identifizieren der Alexa-Session liefert Amazon in der Anfrage eine ID, auf die mittels

```
$alexa_request->session->sessionId
```

zugegriffen werden kann. Mit dieser ID kann eine Session auf dem Server initiierter und eine Session-Variable *Planet* mit dem Wert *Erde* wie folgt belegt werden:

```
session_id($alexa_request->session->sessionId);
session_start();
$_SESSION['Planet']="Erde";
```

Diese Implementierung kann abgewandelt werden, indem wir anstelle der Alexa-SessionID die Alexa-UserID verwenden:

```
$alexa_request->session->user->userId
//oder alternativ
$alexa_request->context->
System->user->userId
```

Die Alexa-UserID ist Session übergreifend gültig. Sie wird beim Aktivieren/Installieren des Skills erzeugt, bis das Skill wieder deaktiviert/deinstalliert wird. Dies bedeutet zum einen, dass nach einer Deaktivierung, gefolgt von einer erneuerten Aktivierung, der Nutzer eine Alexa-UserID für dieses Skill erhält.

Darüber hinaus folgt aus dieser Logik, dass ein und derselbe User in verschiedenen Skills immer eine andere UserID hat und diese nicht Skill-übergreifend genutzt werden kann.

Die Nutzer-Identität in der Alexa Welt ist nicht mit einer Nutzer-Identität der Backend-Applikation verknüpft, welche neben dem Skill zum Beispiel eine Webseite anbieten kann. Um eine solche Verknüpfung herzustellen hat sich ein Workaround etabliert, welcher vorsieht, dass dem Nutzer vom Skill ein individueller PIN-Code angesagt wird, den er auf der zugehörigen Webseite eingeben muss. Alternativ kann auf der Zusammenfassung der Antwort in der Alexa-App (die sogenannte Karte), dem Bildschirm eines Echo Show oder Echo Spot, ein QR-Code mit einem individuellen Link (der zum Beispiel den PIN-Code als Parameter enthält) angezeigt werden.

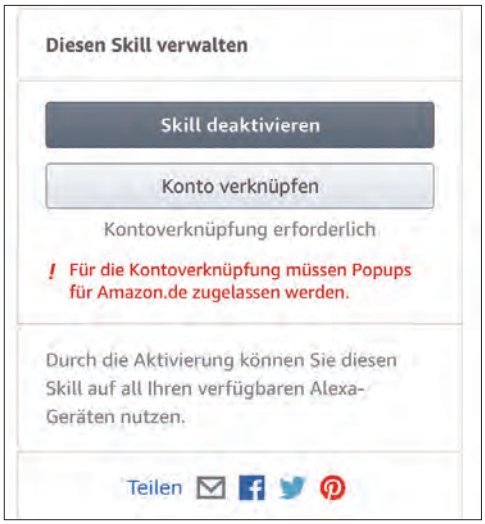
Diese Methode lässt sich nutzen, um über die Webseite mehrere Eingaben vorzunehmen, deren Erfassung per Sprache zu kompliziert beziehungsweise fehleranfällig wäre (zum Beispiel eine E-Mail-Adresse). Über die PIN werden die Daten der richtigen UserID zugeordnet.

Verfügt die Backend-Anwendung bereits über eine Userverwaltung, so kann über die PIN einem eingeloggten Nutzer die Alexa-UserID zugeordnet und dauerhaft genutzt werden. Hierbei muss beachtet werden, dass selbst bei einer 9-stelligen Zufallszahl Kollisionen auftreten können.

Aus diesem Grund sollte die PIN mit der Alexa-UserID in einer Datenbank-Tabelle abgelegt werden, nachdem validiert wurde, dass die PIN nicht bereits vorhanden ist. Zusätzlich sollte ein Zeitstempel mit angelegt und regelmäßig alle Einträge, die älter als zum Beispiel 15 Minuten sind, gelöscht werden.

Direkte Verknüpfung von UserAccounts

Neben diesem Workaround stellt Amazon auch ein eigenes Verfahren zur Verknüpfung einer bestehenden User-Verwaltung zur Verfügung - das sogenannte Account-Linking. Skills, die diese Funktionalität nutzten, werden in der Alexa-App beziehungsweise auf der Amazon-Webseite mit einem Hinweis, dass eine Kontoverknüpfung erforderlich ist, versehen (Bild 1).



Die Konto-Verknüpfung auf der Amazon-Webseite (Bild 2)

Nach der Aktivierung, wird neben dem deaktivierten Button auch ein Button zur Kontoverknüpfung angezeigt (Bild 2).

Klickt der Nutzer auf diesen Button, so wird eine Webseite des Backends geöffnet, auf welcher der Nutzer seine Logendaten eingeben muss.

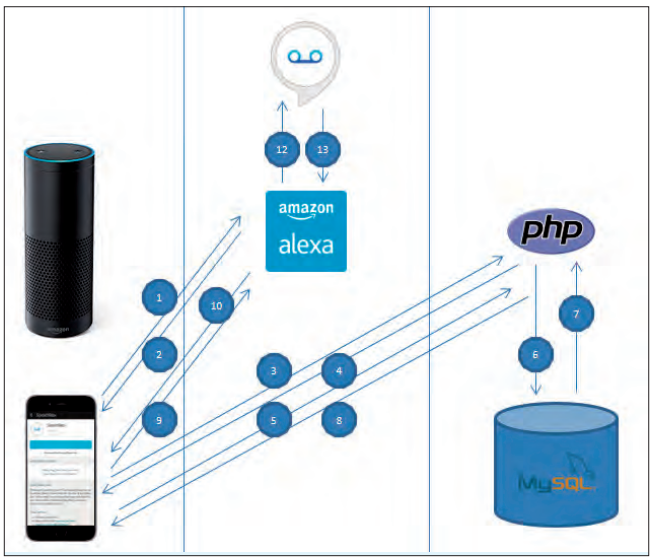
Um diese Funktionalität in einem eigenen Skills zu integrieren, muss diese zuerst in der Skill-Konfiguration unter dem Reiter *Build*, im Menüpunkt *Accountlinking* aktiviert ▶

Alexa Skill Test-Simulator für Entwickler

Für Entwickler stellt Amazon ein Tool für den Test ihrer Alexa-Skills zur Verfügung. Der Alexa Test-Simulator erweitert die vorhandenen Möglichkeiten um Funktionen wie eine visuelle Anzeige, neue Spracheingabefunktionen und die Option, Multi-turn-Gespräche, Entity Resolution, Dialogmanagement und vieles mehr zu testen.

Diese Verbesserungen vereinfachen es Entwicklern, ihre Skills zu testen und optimieren ohne ein Echo-Gerät zu besitzen. Das Tool folgt einer Reihe anderer Features, wie dem Beta Testing Tool, welche dieses Jahr eingeführt wurden, um die Entwicklung ansprechender und qualitativ hochwertiger Skills zu ermöglichen.

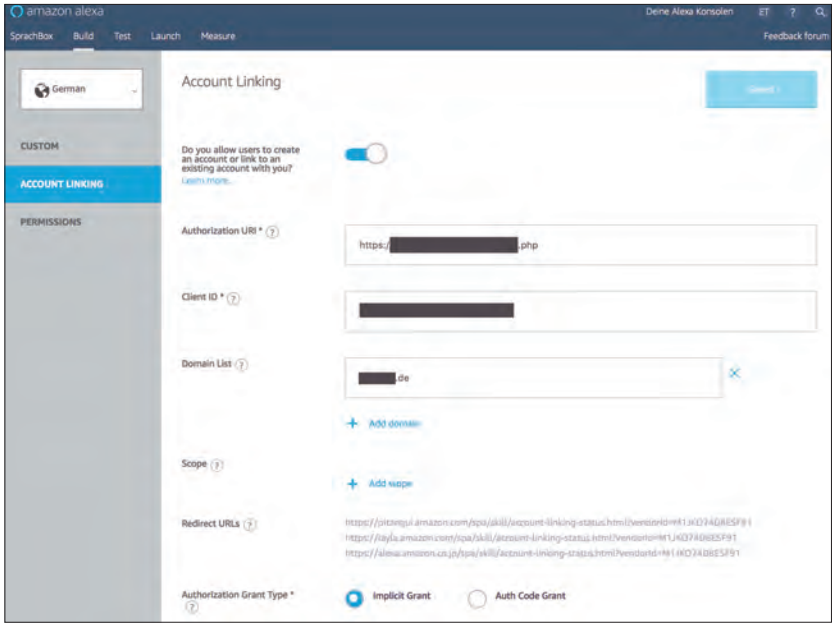
Das Tool unterstützt die Sprachen Englisch-USA, Englisch-UK, Englisch-IN, Englisch-CA, Japanisch und Deutsch.



Ablaufdiagramm des Account-Linking-Prozesses (Bild 4)

werden (Bild 3). Danach erweitert sich die Eingabemaske entsprechend. Die wichtigsten neuen Felder sind:

- **Authorization URI:** Dies ist die URL der bereitgestellten Login Seite. Dabei muss diese Seite per HTTPS erreichbar sein, darf keine weiteren PopUps öffnen und sollte mobil-optimiert sein.
- **Client ID:** Ein Name, mit dem das jeweilige Skill von der Anmeldeseite erkannt werden kann und der als URL Parameter beim Aufruf übergeben wird.
- **Domain List:** Die Alexa-App ist sehr restriktiv, welche Webseiten es öffnet. Verweist die Login-Seite auf eine andere Domain zum Beispiel für eine Video-Anleitung, so muss die Domain hier vorab bekannt gemacht werden. Es können maximal 30 Domains aufgelistet werden.
- **Authorization Grant Type:** In diesem Artikel wird die einfachere Implicit Grant-Option beschrieben, welche im Gegensatz zu Auth Code Grant mit deutlich weniger Aufrufen und der dazu implementierenden Logik auskommt.



Account-Linking Sektion in der Skill-Konfiguration (Bild 3)

Das Zusammenspiel des Login-Skripts mit der Alexa-App und dem Alexa-Skill ist in Bild 4 dargestellt. Dabei erhält die Alexa-App aus der Skill-Definition die URL des Login-Skripts und ruft dieses dann in einem internen Browser auf. Als URL-Parameter werden dabei *state* und *redirect_uri* übergeben, die im späteren Verlauf benötigt werden. Dem ersten Impuls, diese Daten ungeprüft in versteckten Formularfeldern zu übernehmen, sollte nicht nachgegeben werden, da dies potentielle Angriffe ermöglicht.

Empfang der Formulardaten

Wenn man für den Empfang der Formulardaten ein anderes Skript angeben möchte, sollte man die Daten in Session-Variablen schreiben. Einfacher ist es aber dasselbe Skript zu verwenden indem das *action*-Attribut des HTML-Formulars nicht gesetzt wird. Dadurch wird dieselbe URL inklusive der URL-Parameter genutzt.

An dieser Stelle sei noch erwähnt, dass der interne Browser nicht nur - wie erwähnt - Pop-Ups unterdrückt, sondern unter iOS auch keine Universal-Links unterstützt. Wenn die Login-Seite mittels des Button *in Safari Öffnen* an die Safari-App übergeben wird, so werden dort Universal-Links wieder unterstützt, sodass von hier die eigene App direkt gestartet werden kann. Die Password-autofill-Funktion funktioniert allerdings auch im Browser der Alexa App.

Anpassungen vornehmen

Der notwendige Anpassungsaufwand für uns beginnt nach der Rückmeldung der Use-

reingabe. Sollte bei der Überprüfung der Login-Daten ein Fehler auftreten, kann man eine ganz normale HTML-Rückmeldung geben. Nach erfolgreichem Login erwartet Amazon ein sogenanntes AccessToken. Diese Zeichenfolge unterliegt nur der Bedingung, dass wir im Skill damit eindeutig auf den User schließen können, ist ansonsten aber frei konstruierbar. Die Rückmeldung des AccessToken an Amazon erfolgt mittels einer HTTP-Weiterleitung. Diese enthält als Ziel die zuvor erhaltene *redirect_url* und als URL-Parameter das AccessToken, den zuvor erhaltenen *state*, und den Parameter *TokenType*, welcher fest auf *Bearer* gesetzt wird.

Folgende Punkte sind darüber hinaus zu beachten: Die Rückmeldung erfolgt im gleichen Skript, wie die Formularerstellung. Deshalb sind die Amazon-Parameter per $_*GET* verfügbar. Außerdem muss *accessToken* vorher einen eindeutigen String für den User zugewiesen bekommen haben:

```
http_response_code(307);
$protocol = (isset($_SERVER
['SERVER_PROTOCOL']) ? $_SERVER
['SERVER_PROTOCOL'] : 'HTTP/1.0');

header($protocol . ' 307 Temporary
Redirect (since HTTP/1.1)');
header("Location: ".$_GET["redirect_uri"].
"#access_token=".$accessToken."&state=".
$_GET["state"]."&token_type=Bearer");
die();
```

Die Alexa-App führt den Redirekt aus und so wird der *AccessToken* am Alexa-User-Account gespeichert. In einer HTML-Seite erhält der Nutzer eine Rückmeldung über die erfolgreiche Account-Verknüpfung. Anschließend wird bei jedem Skill-Aufruf durch diesen Nutzer neben der UserID im User-Objekt auch das *AccessToken* geliefert. Auf dieses kann wie folgt zugegriffen werden:

```
$alexa_request->session->user->accessToken
```

Oder alternativ so:

```
$alexa_request->context->System->user->accessToken
```

Als Besonderheit gilt es zu beachten, dass ein Skill mit Account-Linking immer eine Aufforderung zur Account-Verknüpfung schicken muss, wenn der *AccessToken* nicht vorhanden oder ungültig ist (Listing 1).

Auch wenn das AccessToken nicht für jede Interaktion benötigt wird - beim *LaunchRequest* oder *AMAZON.HelpIntent* könnte zum Beispiel auch eine allgemeine Information ausgegeben werden - so prüft Amazon die Einhaltung dieser Vorgabe dennoch bei der Freigabe des Skills sehr strikt.

Listing 1: Check des AccessToken

```
{
   "version": "1.0",
   "response": {

      "outputSpeech": {
         "type": „PlainText“,
         "text": "Um das Skill XYZ nutzen zu können,
         muss es mit einem XYZ Account verbunden
         werden."
        },

      "card": {
         "type": "LinkAccount"
      },
   "shouldEndSession": true
   },
   "sessionAttributes": {}
}
```

Links zum Thema

- Amazon Developer Portal: Request and Response JSON Reference
  https://developer.amazon.com/de/docs/custom-skills/request-and-response-json-reference.html
- Amazon Developer Portal: SessionEndedRequest
  https://developer.amazon.com/de/docs/custom-skills/request-types-reference.html#sessionendedrequest
- Amazon Developer Portal: Verknüpfen eines Alexa-Benutzers mit einem Benutzer in Ihrem System
  https://developer.amazon.com/de/docs/custom-skills/link-an-alexa-user-with-a-user-in-your-system.html

Fazit

Ein allgemeines Skill lässt sich mit wenigen Anpassungen in ein personalisiertes Skill verwandeln und mit einer eigenen Nutzerverwaltung integrieren. Wer darüber hinaus noch mehr Individualisierung wünscht, kann zusätzlich auch über das durch den Nutzer verwendete Endgerät differenzieren. Dazu muss nur

```
$alexa_request->context->System->device->deviceId
```

ausgewertet werden. Dabei gilt es aber zu prüfen, ob diese Differenzierung aus Nutzersicht auch Sinn ergibt. ■

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