I'd been struggling for weeks with proper syntax for using session attributes in a skill because I've never worked with them before, and the sample AWS FavoriteColors app uses so many duplicate naming conventions that it's difficult for a session attributes noob to tease out the necessary steps to follow in his or her own code. Having finally gotten it right, with the help of some devs on the Amazon Alexa/Echo dev board (Greg Crawford, Greg Laabs, James Chivers - thanks so much!) I wanted to share what I've learned. Again, I'm a session attributes noob so there may be better ways to do or explain what's covered here; I'm just sharing what finally worked for me to get a session attribute to work in my skill.
- April

1. By default the Echo doesn't "remember" things between server calls, even in the same session.
So if you have one function that sets a counter, and you need to increment that counter every time another function runs in the same session, you will have to use a session attribute to first define/declare the counter in the first function and then to increment it in the other function.

2. There are numerous references to session attributes in many of the Amazon-created skills.
If you're starting with one of those scripts/skills as your template, leave the early references to session attributes (those that come before the "functions that control the app's behavior" part of the script) as-is.

3. Ensure your buildResponse function includes the sessionAttributes line shown below.
This is the response object that will "remember" your session attribute(s) between functions/responses running in the same session.

```
function buildResponse(sessionAttributes, speechletResponse) {
return {
      version: "1.0",
      sessionAttributes: sessionAttributes,
      response: speechletResponse
      };
}
```

4. Session attributes are declared, stored and passed as JSON name: value pairs.
They must be declared/defined as such for the server to recognize them as JSON objects to be passed to and from the server in the session.

5. Here's the correct syntax to declare/define a session attribute.
First you declare a variable to hold your session attributes, defining it with curly braces so the server will know it's a JSON object. Here I'm using the same variable name as in Amazon's sample skills - sessionAttributes. Then you assign one (or more) name: value pairs to that sessionAttributes variable. The single session attributes variable can contain more than one session attribute because the attributes object of the session is made up of name: value pairs; you can think of it like an array object, if that clarifies things at all.

```
var sessionAttributes = {};
```

---[code needed to calculate/create the value you will save as a session attribute]---

```
sessionAttributes = {nameOfThisSessionAttribute: valueOfThisSessionAttribute};
```

Note that the valueOfThisSessionAttribute can be a literal value, a variable that's calculated by the preceding code, an array or array reference, a function reference, etc. All the usual options you'd expect in javascript are there.

If you want to declare multiple session attribute name: value pairs at once, you'd do it like this:

```
sessionAttributes = {
nameOfPairOne: valueOfPairOne,
nameOfPairTwo: valueOfPairTwo,
nameOfPairThree: valueOfPairThree,
};
```

Notice how this is the same construction as that used in the buildResponse function shown on the previous page, and in the Intent Schema for your skill. All of these are declaring name: value pairs in JSON notation.

6. To reference a session attribute previously declared/defined,
first declare a local variable in the function where the session attribute value will be needed, at the start of the function, and set it equal to the attributes object of the session, like this:

```
var sessionAttributes = session.attributes;
```

To reference any session attributes you've previously set, which are now exposed in the sessionAttributes variable, use this notation:

```
sessionAttributes.nameOfAttributeFromNameValuePair
```

Now you can call or manipulate the value associated with the name you assigned in the name: value pair. For example, to change the value of the attribute you could do this, where newValue is a literal value, a variable calculated prior to this line of code, a value returned from another function, etc. etc.

```
sessionAttributes.nameOfAttributeFromNameValuePair = newValue;
```

6. (cont'd)...to reference the attribute for use in calculations, speech responses, etc. you can set a local variable equal to it and use that variable for calculations, spoken responses, etc., like this:

```
var myLocalVariable = sessionAttributes.nameOfAttribute;
```

----

7. The callback at the end of any function in your script where a session attribute is declared or referenced
must include the sessionAttributes variable you've set, like this:

```
callback(sessionAttributes,
buildSpeechletResponse(intent.name, speechOutput, repromptText, shouldEndSession));
```

8. If your session attributes don't seem to be behaving as expected,
double-check their declaration/reference locations in your code. Session attributes will be treated as undefined values in a given function until they've been exposed to that function by reference, as explained in step 6 above.

9. If your skill involves a routine/game the user can re-start in the same session,
such as a 'guess the number' game, don't forget to clear any previously-set session attributes in the function that starts the routine/game.

This requirement may mean you have to break out the first round of your routine/game as a separate function in the session (probably separate from the welcomeResponse function too, unless you want to run the welcomeResponse every time the user starts a new routine/game in the same session), then have another function that handles all the routine/game processing from that point on, to the end of the routine/game.

In the start routine/game function, you can clear all previously-set attributes with these declarations at the start of the function:

```
var sessionAttributes = {};
session.attributes = sessionAttributes;
```

You can clear a specific attribute, but still keep the name: value pair in the session, like this:

```
var sessionAttributes = session.attributes
sessionAttributes.nameOfAttributeToClear = "";
```

10. I found it helpful to simply declare the sessionAttributes = {}; at the top of every function.
This coding convention made it easy for me to see at a glance while scanning my code where and when I'd included session attribute references, and any needed session attribute references that were missing from a given function in my code jumped right out at me.