**Alexa APL**

**Text and speech – using Transformers**

**My video:** [**https://youtu.be/x6425OyqD8E**](https://youtu.be/x6425OyqD8E)

Recommended watching:

https://www.youtube.com/watch?v=IGsvRT\_Ak\_M

<https://developer.amazon.com/en-GB/blogs/alexa/alexa-skills-kit/2020/05/how-to-build-a-slideshow-project-with-alexa-presentation-language?ref=tsm_1_tw_s__3337144833&linkId=88441015>

**Let’s start:**

**Create a new Alexa hosted skill**

add Alexa Presentation Language, choose

**interfaces > APL > save interfaces**

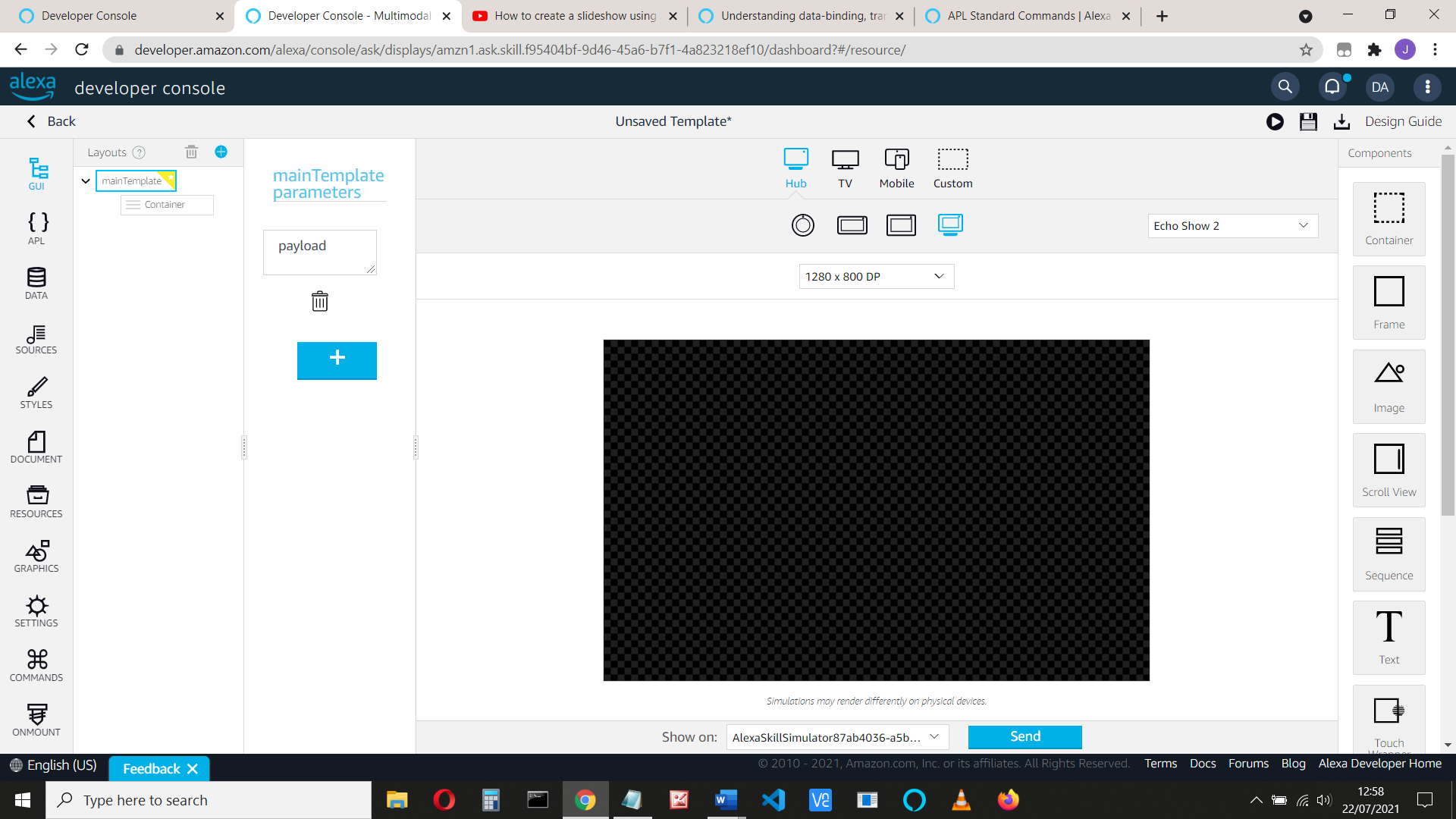
**rebuild model**

Create the display using the GUI:

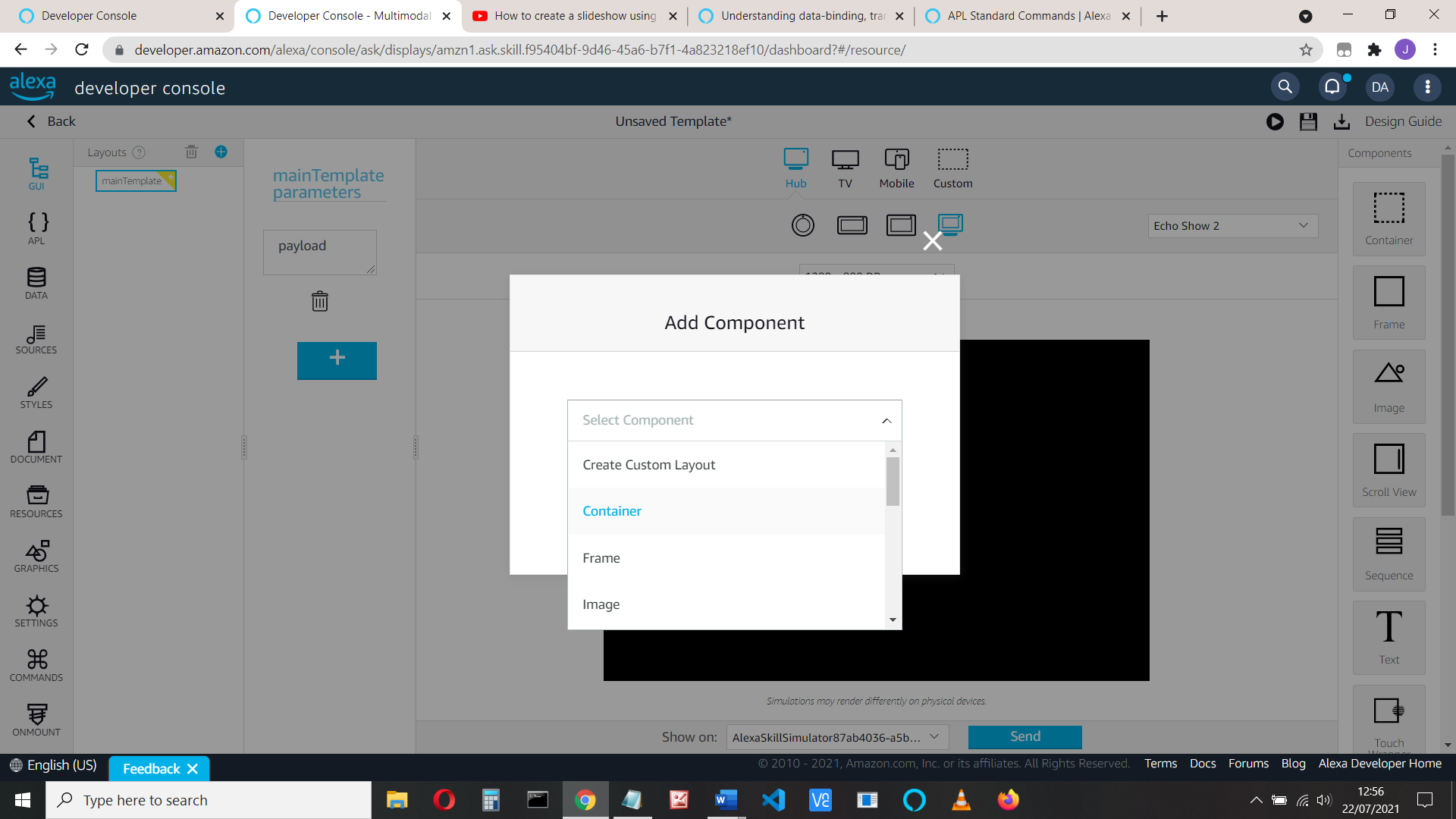
Select Build tab:

Select Multimodal responses > visual > create visual response > blank document

Click the + icon just to the top right of ‘mainTemplate’,



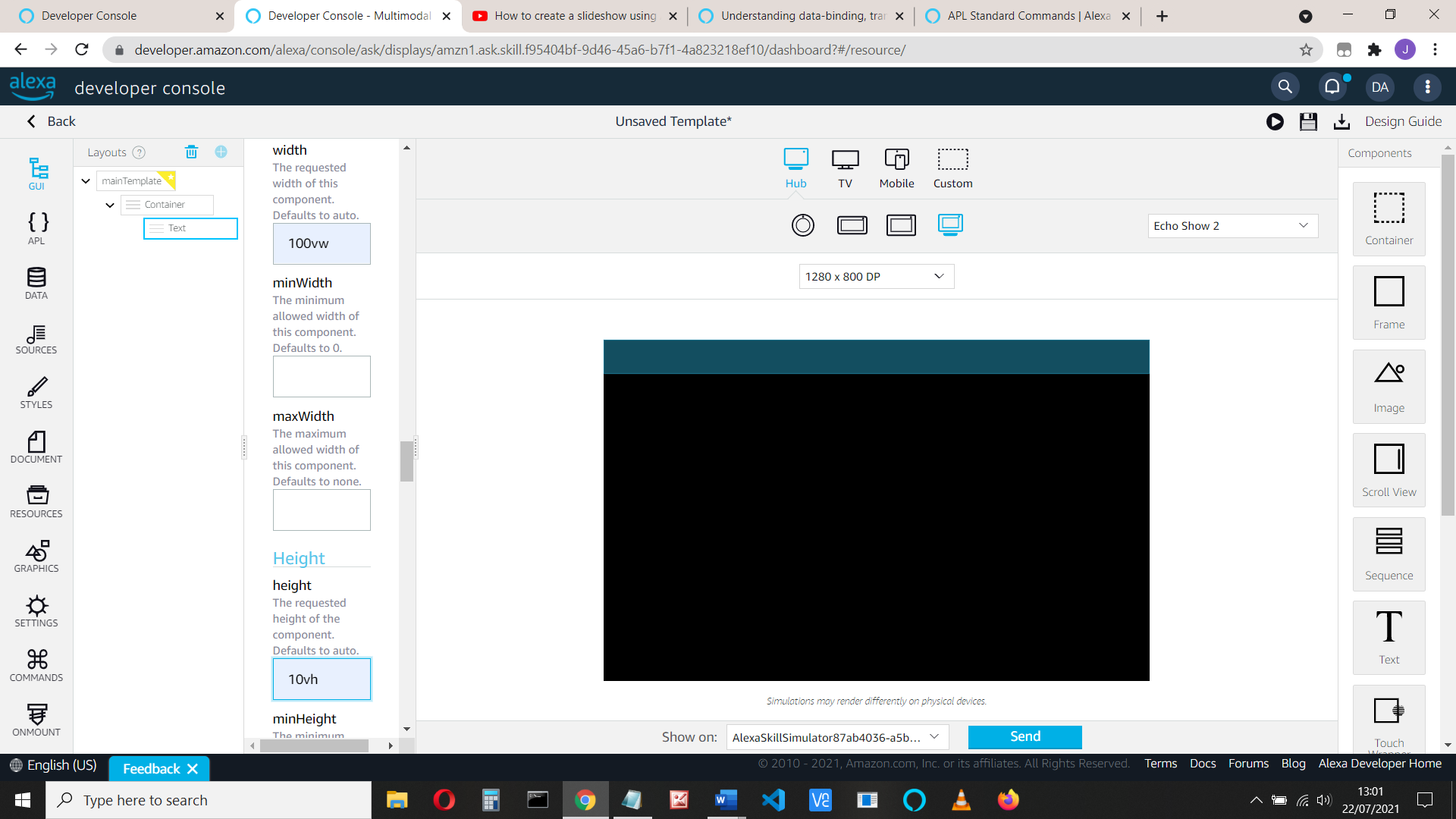
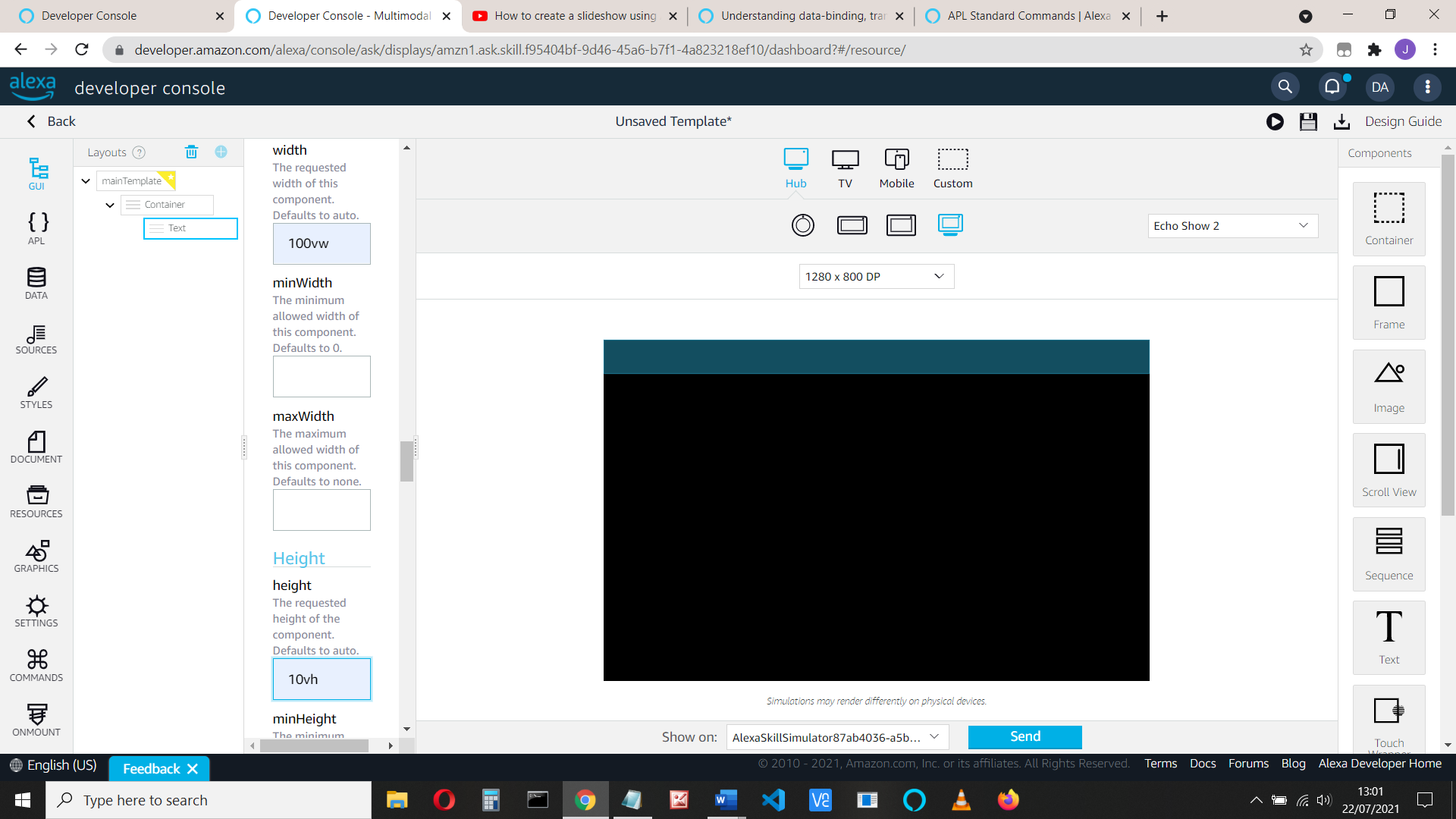
and add a component - Container



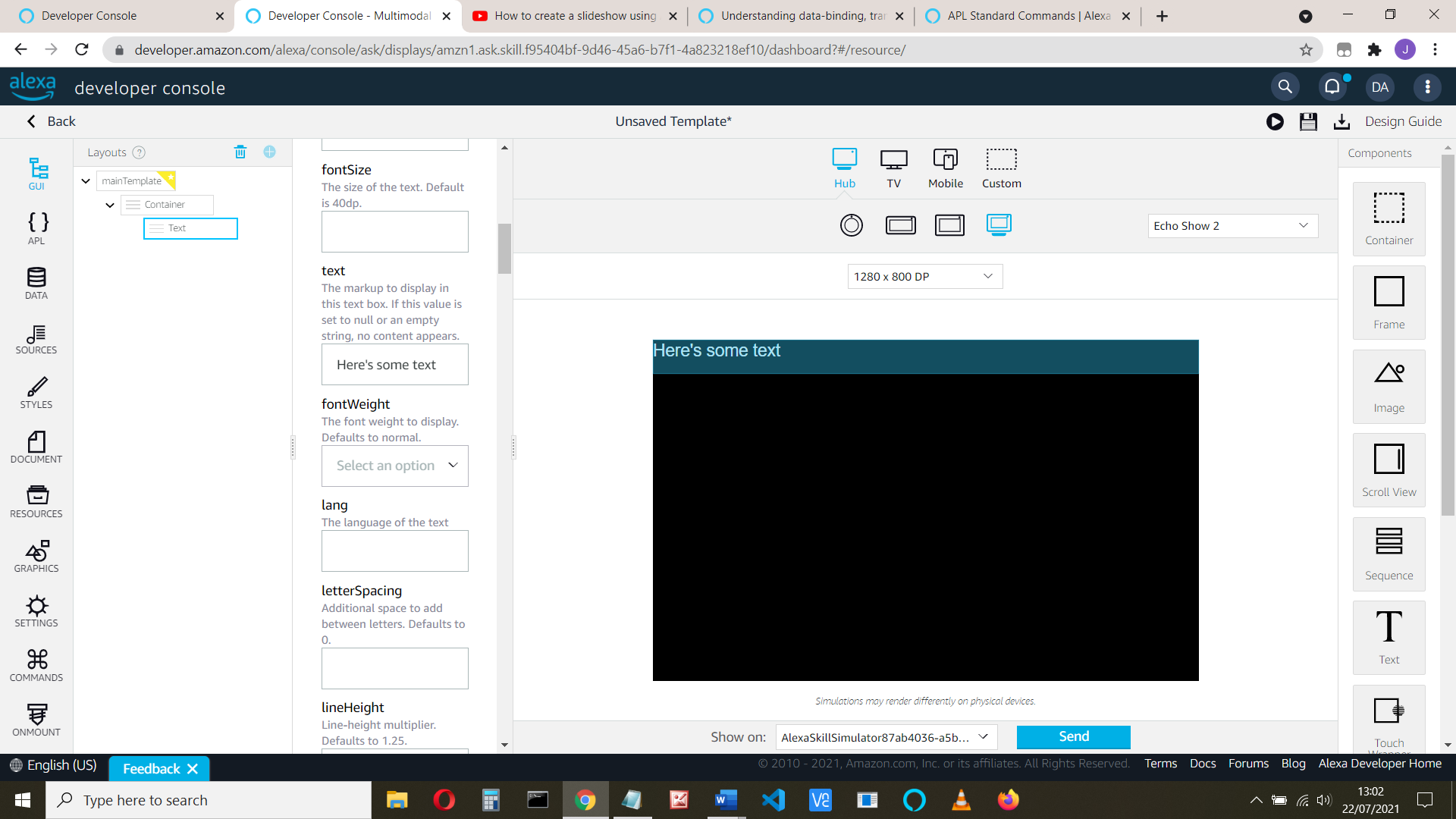
Click the container you just added and add a text component within the Container

We’ll make the text use the full width abut 10vh high:

Scroll down through the text properties and make the height and width changes as shown:



And add some text in the text field to show that it displays:



Let’s see how we can get the text form our data

Click the data icon and add the following text:

{

"data": {

"properties": {

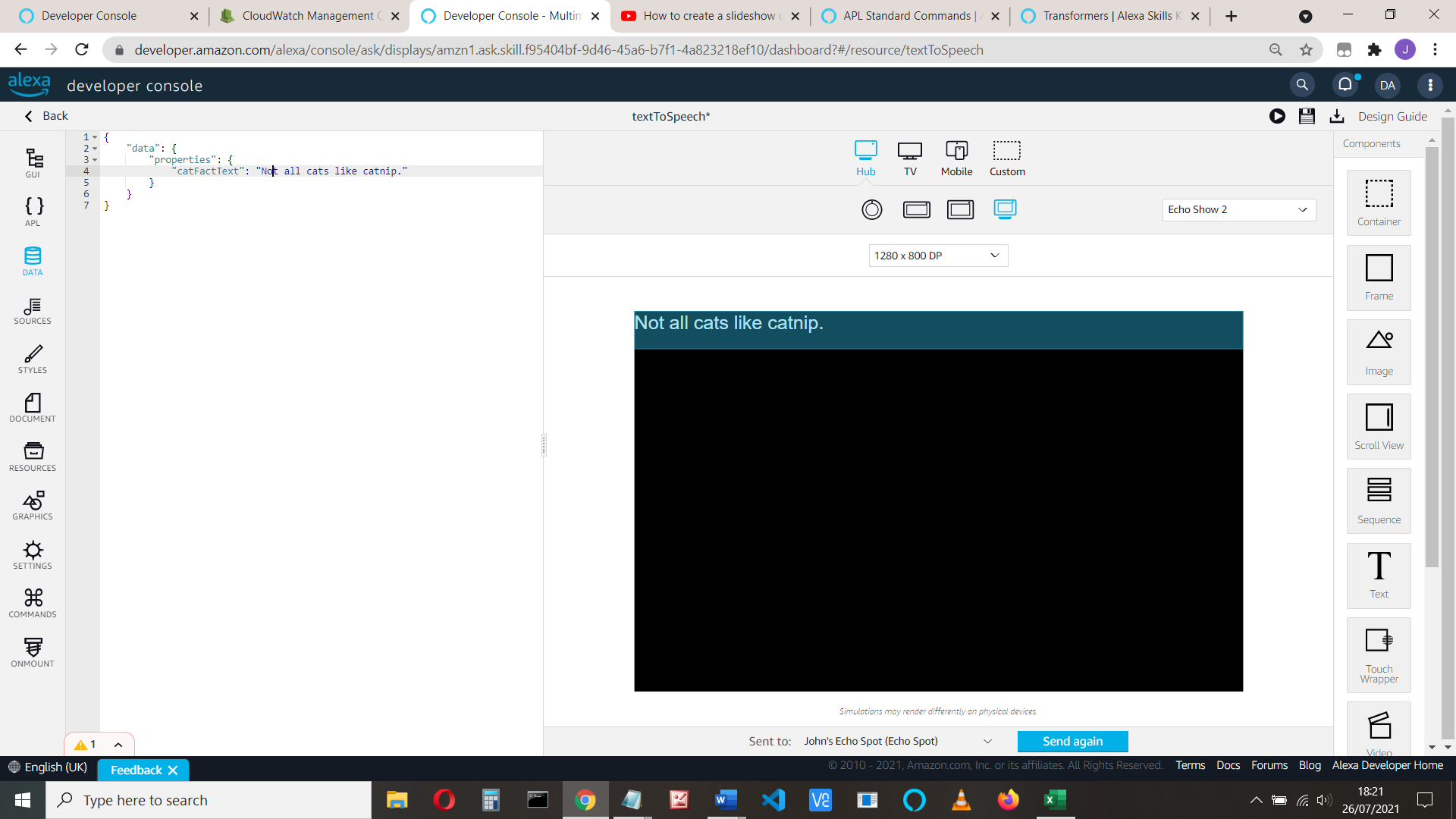
"catFactText": "Not all cats like catnip."

}

}

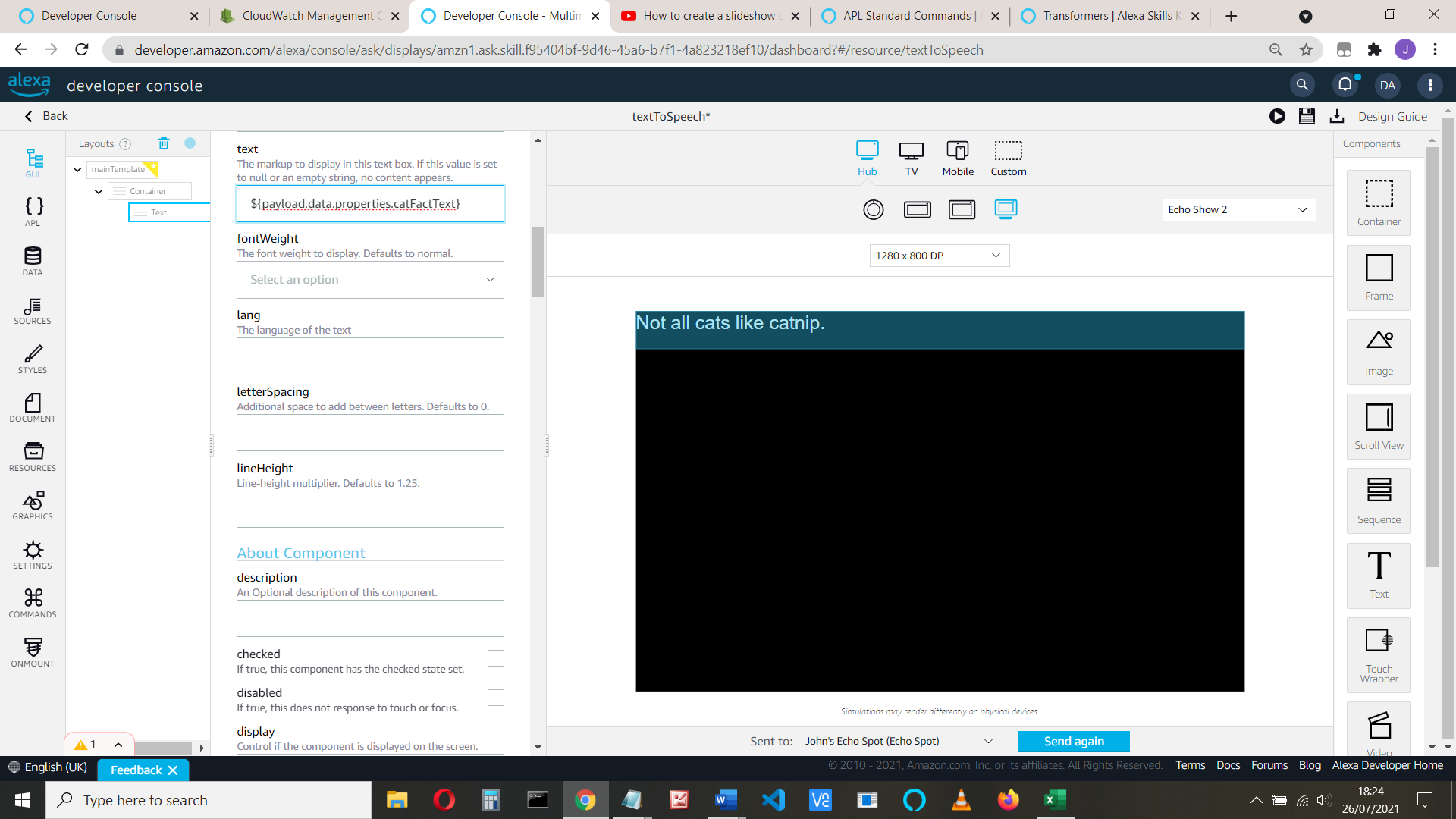
}

It looks like this:



Return to GUI icon and change the container component’s data to

${payload.data.properties.catFactText}



Don’t worry about the red underlining

Now we want to say the text when the text is loaded

We do this using a transformer which in our case will transform the text to the speech (we could also transform ssml to speech or other such as aplAudioToSpeech, ssmlToText or textToHint.

The transformer takes the text from an inputPath, transforms it and sends it to an outputName. It’s defined as follows:

"transformers": [

{

"inputPath": "catFactText",

"outputName": "outputSpeech",

"transformer": "textToSpeech"

}

]

Add this to the data so that it’s like this:

{

"data": {

"properties": {

"catFactText": "Not all cats like catnip."

}

},

"transformers": [

{

"inputPath": "catFactText":

"outputName": "outputSpeech",

"transformer": "textToSpeech"

}

]

}

BE CAREFUL – you may have a working json, but it may not work: e.g.:

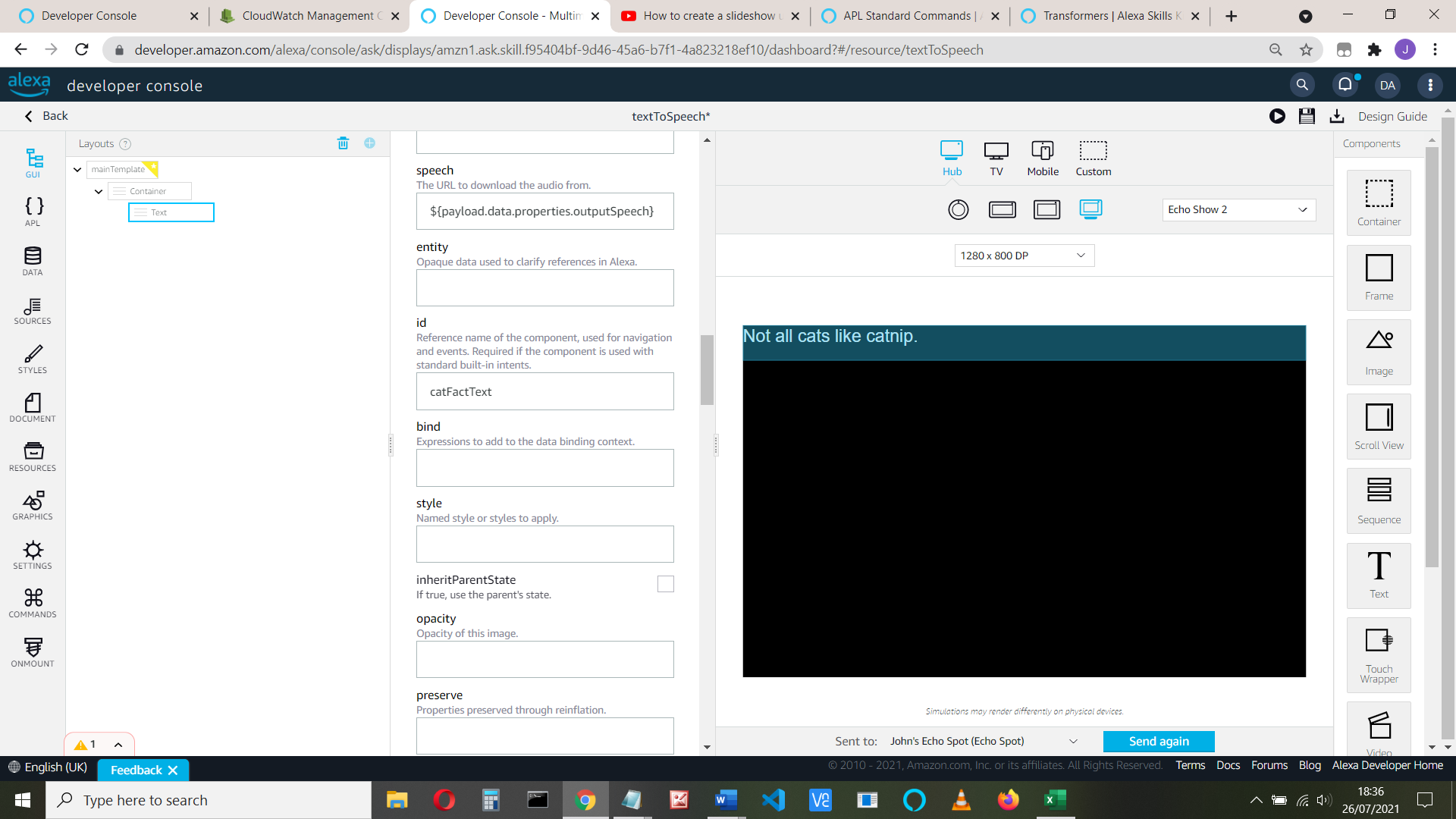
|  |  |
| --- | --- |
| This works:  {  "data": {  "properties": {  "catFactText": "Not all cats like catnip."  },  "transformers": [  {  "inputPath": "catFactText",  "outputName": "outputSpeech",  "transformer": "textToSpeech"  }  ]  }  } | This fails:  {  "data": {  "properties": {  "catFactText": "Not all cats like catnip."  }  },  "transformers": [  {  "inputPath": "catFactText",  "outputName": "outputSpeech",  "transformer": "textToSpeech"  }  ]  } |

The Text component has a **speech** property that is used to say the words required. This comes from the outputName of the transform, the input from the inputPath: this is the id of the Text component. Add the following to the text component, either in the GUI or in the APl editor

(editor:) “speech": "${payload.data.properties.outputSpeech}",

"id": "catFactText",

(GUI:)



We need an event for this to occur. We’ll use OnMount. We can either add this in the APL, or in the code (See the end on how to do this). We’ll start by adding in the APL.

The SpeakItem needs to know which item it is speaking. This is done using the SpeakItem’s componentId matching with the text component id

This is easiest in the editor.

Change the code to the following

{

"type": "APL",

"version": "1.6",

"license": "Copyright 2021 Amazon.com, Inc. or its affiliates.",

"settings": {},

"theme": "dark",

"import": [],

"resources": [],

"styles": {},

"onMount": [],

"graphics": {},

"commands": {},

"layouts": {},

"mainTemplate": {

"parameters": [

"payload"

],

"items": [

{

"items": [

{

"text": "${payload.data.properties.catFactText}",

"type": "Text",

"speech": "${payload.data.properties.outputSpeech}",

"id": "catFactText",

**"onMount": [**

**{**

**"type": "SpeakItem",**

**"componentId": "catFactText"**

**}**

**],**

"width": "100vw",

"height": "10vh"

}

],

"type": "Container"

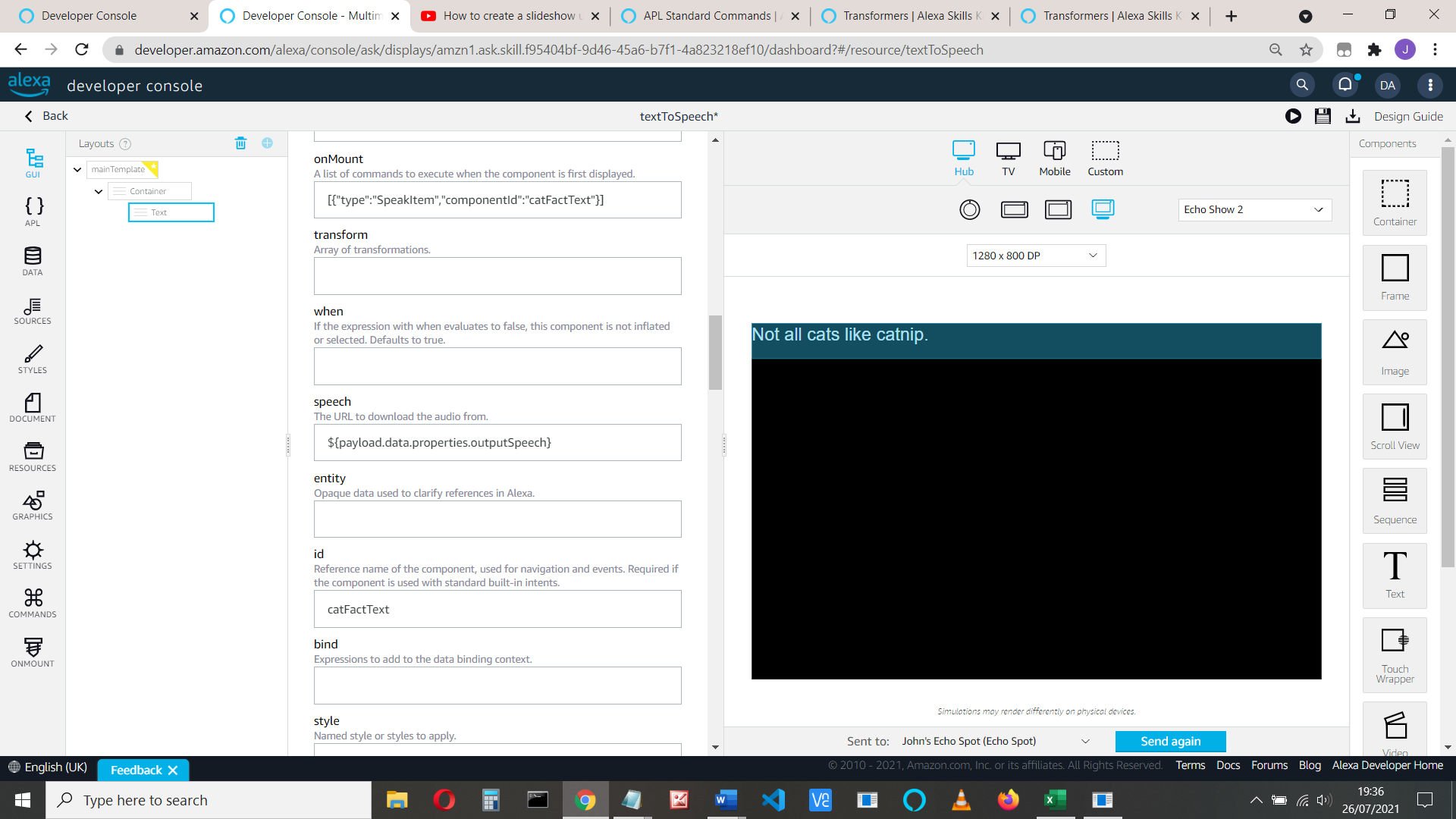
}

]

}

}

Of course, this adds it to the GUI:



If you want to do this in the python code (rather than the json), you’ll need to add an ExecuteCommands directive with the speakItem command. We’ll see that later (maybe)

Ok try it out. Click the Play button (preview mode) in the top right-hand corner by ‘Design Guide’

Now let’s create a skill that uses this.

**Create a new Alexa hosted skill (Python)**

**add** Alexa Presentation Language, choose

**interfaces > APL > save interfaces**

**rebuild model**

Click code tab and add the following to the code, just before *class LaunchRequestHandler*:

**from ask\_sdk\_core.utils import get\_supported\_interfaces**

**import json**

**from ask\_sdk\_model.interfaces.alexa.presentation.apl import (**

**RenderDocumentDirective, ExecuteCommandsDirective, SpeakItemCommand, AutoPageCommand, HighlightMode)**

**def \_load\_apl\_document(file\_path):**

**# type: (str) -> Dict[str, Any]**

**"""Load the apl json document at the path into a dict object."""**

**with open(file\_path) as f:**

**return json.load(f)**

Replace launch handler code with:

**class LaunchRequestHandler(AbstractRequestHandler):**

**"""Handler for Skill Launch."""**

**def can\_handle(self, handler\_input):**

**# type: (HandlerInput) -> bool**

**return ask\_utils.is\_request\_type("LaunchRequest")(handler\_input)**

**def handle(self, handler\_input):**

**# type: (HandlerInput) -> Response**

**# speak\_output = "Hello, you can say Hello or Help. Which would you like to try?"**

**if get\_supported\_interfaces(handler\_input).alexa\_presentation\_apl is not None:**

**return (**

**handler\_input.response\_builder**

**#.speak(speak\_output)**

**.ask("Keep the session open")**

**.add\_directive(**

**RenderDocumentDirective(**

**token="pagerToken",**

**document=\_load\_apl\_document("main.json"),**

**datasources=\_load\_apl\_document("data.json")**

**)**

**)**

**.response**

**)**

**else:**

**speak\_output = "You don't have a screen"**

**return (**

**handler\_input.response\_builder**

**.speak(speak\_output)**

**.ask("Please buy an Alexa device with a screen")**

**.response**

**)**

**Add the data.json**

{

"data": {

"properties": {

"catFactText": "Not all cats like catnip."

},

"transformers": [{

"inputPath": "catFactText",

"outputName": "outputSpeech",

"transformer": "textToSpeech"

}]

}

}

**And add main.json**

{

"type": "APL",

"version": "1.6",

"license": "Copyright 2021 Amazon.com, Inc. or its affiliates.",

"settings": {},

"theme": "dark",

"import": [],

"resources": [],

"styles": {},

"onMount": [],

"graphics": {},

"commands": {},

"layouts": {},

"mainTemplate": {

"parameters": [

"payload"

],

"items": [

{

"items": [

{

"text": "${payload.data.properties.catFactText}",

"type": "Text",

"speech": "${payload.data.properties.outputSpeech}",

"id": "catFactText",

"onMount": [

{

"type": "SpeakItem"

},

{

"componentId": "myId"

}

],

"width": "100vw",

"height": "10vh"

}

],

"type": "Container"

}

]

}

}

Execute the skill.

**Using ExecuteDirective:**

Remove the OnMount from main.json, so that it is:

{

"type": "APL",

"version": "1.6",

"license": "Copyright 2021 Amazon.com",

"settings": {},

"theme": "dark",

"import": [],

"resources": [],

"styles": {},

"onMount": [],

"graphics": {},

"commands": {},

"layouts": {},

"mainTemplate": {

"parameters": [

"payload"

],

"items": [

{

"type": "Container",

"items": [

{

"text": "${payload.data.properties.catFactText}",

"type": "Text",

"width": "100vw",

"height": "10vh",

"speech": "${payload.data.properties.outputSpeech}",

"id": "catFactText"

}

]

}

]

}

}

**Change the LanchRequestHandler to include the ExecuteCommandsDirective**

class LaunchRequestHandler(AbstractRequestHandler):

"""Handler for Skill Launch."""

def can\_handle(self, handler\_input):

# type: (HandlerInput) -> bool

return ask\_utils.is\_request\_type("LaunchRequest")(handler\_input)

def handle(self, handler\_input):

# type: (HandlerInput) -> Response

# speak\_output = "Hello, you can say Hello or Help. Which would you like to try?"

if get\_supported\_interfaces(handler\_input).alexa\_presentation\_apl is not None:

return (

handler\_input.response\_builder

#.speak(speak\_output)

.ask("Keep the session open")

.add\_directive(

RenderDocumentDirective(

token="pagerToken",

document=\_load\_apl\_document("main.json"),

datasources=\_load\_apl\_document("data.json")

)

)

**.add\_directive(**

**ExecuteCommandsDirective(**

**token="pagerToken",**

**commands = [{**

**"type" : "SpeakItem",**

**"componentId" : "catFactText"**

**}]**

**)**

**)**

.response

)

else:

speak\_output = "You don't have a screen"

return (

handler\_input.response\_builder

.speak(speak\_output)

.ask("Please buy an Alexa device with a screen")

.response

)

Try it!

**Part 2 – Adding an image**

I’ve loaded an image to my s3 bucket at

<https://s3.eu-west-1.amazonaws.com/johnallwork.com/photos/perkin.jpg>

You can use the S3 bucket associated with this Alexa Hosted Code

Return to GUI design. Add an image with the properties:

"source": "https://s3.eu-west-1.amazonaws.com/johnallwork.com/photos/perkin.jpg",

"type": "Image",

"id": "perkinId",

"speech": "${payload.data.properties.outputSpeech}",

"width": "100vw",

"height": "90vh",

Note if you want the image above the text, just drag the GUI

The APL is now:

{

"type": "APL",

"version": "1.6",

"license": "Copyright 2021 Amazon.com",

"settings": {},

"theme": "dark",

"import": [],

"resources": [],

"styles": {},

"onMount": [],

"graphics": {},

"commands": {},

"layouts": {},

"mainTemplate": {

"parameters": [

"payload"

],

"items": [

{

"type": "Container",

"items": [

{

"text": "${payload.data.properties.catFactText}",

"type": "Text",

"width": "100vw",

"height": "10vh",

"speech": "${payload.data.properties.outputSpeech}",

"id": "catFactText"

},

{

"source": "https://s3.eu-west-1.amazonaws.com/johnallwork.com/photos/perkin.jpg",

"type": "Image",

"id": "perkinId",

"speech": "${payload.data.properties.outputSpeech}",

"width": "100vw",

"height": "90vh",

"onMount": [

{

"type": "SpeakItem",

"componentId": "perkinId"

}

]

}

]

}

]

}

}

And change the transformer inputPath and add the perkinText to the data json so that it becomes:

{

"data": {

"properties": {

"catFactText": "Not all cats like catnip.",

"perkinText": "This is perkin"

},

"transformers": [

{

"inputPath": "perkinText",

"outputName": "outputSpeech",

"transformer": "textToSpeech"

}

]

}

}

Cleck Preview mode to hear it say "This is perkin".

Now, if you haven’t done so, watch the Alexa evangelist video at:

https://www.youtube.com/watch?v=IGsvRT\_Ak\_M

That will tell you more about creating components and data binding.