



# **Customize States and Flows for Retail Use Case**

NetApp Solutions

Kevin Hoke  
May 24, 2021

This PDF was generated from [https://docs.netapp.com/us-en/netapp-solutions/ai/cainvidia\\_customize\\_states\\_and\\_flows\\_for\\_retail\\_use\\_case.html](https://docs.netapp.com/us-en/netapp-solutions/ai/cainvidia_customize_states_and_flows_for_retail_use_case.html) on October 21, 2021. Always check docs.netapp.com for the latest.

# Table of Contents

- Customize States and Flows for Retail Use Case ..... 1
  - main\_flow.yml ..... 1
  - retail\_flow.yml ..... 5
  - weather\_flow.yml ..... 6
  - error\_flow.yml ..... 8

# Customize States and Flows for Retail Use Case

You can customize States and Flows of Dialog Manager for your specific use cases. In our retail example, we have the following four yaml files to direct the conversation according to different intents.

See the following list of file names and description of each file:

- `main_flow.yml`: Defines the main conversation flows and states and directs the flow to the other three yaml files when necessary.
- `retail_flow.yml`: Contains states related to retail or points-of-interest questions. The system either provides the information of the nearest store, or the price of a given item.
- `weather_flow.yml`: Contains states related to weather questions. If the location cannot be determined, the system asks a follow up question to clarify.
- `error_flow.yml`: Handles cases where user intents do not fall into the above three yaml files. After displaying an error message, the system re-routes back to accepting user questions. The following sections contain the detailed definitions for these yaml files.

## main\_flow.yml

```
name: JarvisRetail
intent_transitions:
  jarvis_error: error
  price_check: retail_price_check
  inventory_check: retail_inventory_check
  store_location: retail_store_location
  weather.weather: weather
  weather.temperature: temperature
  weather.sunny: sunny
  weather.cloudy: cloudy
  weather.snow: snow
  weather.rainfall: rain
  weather.snow_yes_no: snowfall
  weather.rainfall_yes_no: rainfall
  weather.temperature_yes_no: tempyesno
  weather.humidity: humidity
  weather.humidity_yes_no: humidity
  navigation.startnavigationpoi: retail # Transitions should be context
and slot based. Redirecting for now.
  navigation.geteta: retail
  navigation.showdirection: retail
  navigation.showmappoi: idk_what_you_talkin_about
  nomatch.none: idk_what_you_talkin_about
states:
  init:
```

```

    type: message_text
    properties:
      text: "Hi, welcome to NARA retail and weather service. How can I
help you?"
    input_intent:
      type: input_context
    properties:
      nlp_type: jarvis
      entities:
        intent: dontcare
# This state is executed if the intent was not understood
dont_get_the_intent:
  type: message_text_random
  properties:
    responses:
      - "Sorry I didn't get that! Please come again."
      - "I beg your pardon! Say that again?"
      - "Are we talking about weather? What would you like to know?"
      - "Sorry I know only about the weather"
      - "You can ask me about the weather, the rainfall, the
temperature, I don't know much more"
    delay: 0
    transitions:
      next_state: input_intent
idk_what_you_talkin_about:
  type: message_text_random
  properties:
    responses:
      - "Sorry I didn't get that! Please come again."
      - "I beg your pardon! Say that again?"
      - "Are we talking about retail or weather? What would you like to
know?"
      - "Sorry I know only about retail and the weather"
      - "You can ask me about retail information or the weather, the
rainfall, the temperature. I don't know much more."
    delay: 0
    transitions:
      next_state: input_intent
error:
  type: change_context
  properties:
    update_keys:
      intent: 'error'
  transitions:
    flow: error_flow
retail_inventory_check:

```

```
    type: change_context
    properties:
      update_keys:
        intent: 'retail_inventory_check'
    transitions:
      flow: retail_flow
retail_price_check:
  type: change_context
  properties:
    update_keys:
      intent: 'check_item_price'
  transitions:
    flow: retail_flow
retail_store_location:
  type: change_context
  properties:
    update_keys:
      intent: 'find_the_store'
  transitions:
    flow: retail_flow
weather:
  type: change_context
  properties:
    update_keys:
      intent: 'weather'
  transitions:
    flow: weather_flow
temperature:
  type: change_context
  properties:
    update_keys:
      intent: 'temperature'
  transitions:
    flow: weather_flow
rainfall:
  type: change_context
  properties:
    update_keys:
      intent: 'rainfall'
  transitions:
    flow: weather_flow
sunny:
  type: change_context
  properties:
    update_keys:
      intent: 'sunny'
```

```
    transitions:
      flow: weather_flow
cloudy:
  type: change_context
  properties:
    update_keys:
      intent: 'cloudy'
  transitions:
    flow: weather_flow
snow:
  type: change_context
  properties:
    update_keys:
      intent: 'snow'
  transitions:
    flow: weather_flow
rain:
  type: change_context
  properties:
    update_keys:
      intent: 'rain'
  transitions:
    flow: weather_flow
snowfall:
  type: change_context
  properties:
    update_keys:
      intent: 'snowfall'
  transitions:
    flow: weather_flow
tempyesno:
  type: change_context
  properties:
    update_keys:
      intent: 'tempyesno'
  transitions:
    flow: weather_flow
humidity:
  type: change_context
  properties:
    update_keys:
      intent: 'humidity'
  transitions:
    flow: weather_flow
end_state:
  type: reset
```

```
transitions:
  next_state: init
```

## retail\_flow.yml

```
name: retail_flow
states:
  store_location:
    type: conditional_exists
    properties:
      key: '{{location}}'
    transitions:
      exists: retail_state
      notexists: ask_retail_location
  retail_state:
    type: Retail
    properties:
    transitions:
      next_state: output_retail
  output_retail:
    type: message_text
    properties:
      text: '{{retail_status}}'
    transitions:
      next_state: input_intent
  ask_retail_location:
    type: message_text
    properties:
      text: "For which location? I can find the closest store near you."
    transitions:
      next_state: input_retail_location
  input_retail_location:
    type: input_user
    properties:
      nlp_type: jarvis
      entities:
        slot: location
        require_match: true
    transitions:
      match: retail_state
      notmatch: check_retail_jarvis_error
  output_retail_acknowledge:
    type: message_text_random
    properties:
      responses:
```

```

    - 'ok in {{location}}'
    - 'the store in {{location}}'
    - 'I always wanted to shop in {{location}}'
  delay: 0
  transitions:
    next_state: retail_state
output_retail_notlocation:
  type: message_text
  properties:
    text: "I did not understand the location. Can you please repeat?"
  transitions:
    next_state: input_intent
check_rerail_jarvis_error:
  type: conditional_exists
  properties:
    key: '{{jarvis_error}}'
  transitions:
    exists: show_retail_jarvis_api_error
    notexists: output_retail_notlocation
show_retail_jarvis_api_error:
  type: message_text
  properties:
    text: "I am having troubled understanding right now. Come again on that?"
  transitions:
    next_state: input_intent

```

## weather\_flow.yml

```

name: weather_flow
states:
  check_weather_location:
    type: conditional_exists
    properties:
      key: '{{location}}'
    transitions:
      exists: weather_state
      notexists: ask_weather_location
  weather_state:
    type: Weather
    properties:
      transitions:
        next_state: output_weather
  output_weather:
    type: message_text

```



```

    properties:
      text: '{{weather_status}}'
    transitions:
      next_state: input_intent
ask_weather_location:
  type: message_text
  properties:
    text: "For which location?"
  transitions:
    next_state: input_weather_location
input_weather_location:
  type: input_user
  properties:
    nlp_type: jarvis
    entities:
      slot: location
      require_match: true
  transitions:
    match: weather_state
    notmatch: check_jarvis_error
output_weather_acknowledge:
  type: message_text_random
  properties:
    responses:
      - 'ok in {{location}}'
      - 'the weather in {{location}}'
      - 'I always wanted to go in {{location}}'
    delay: 0
  transitions:
    next_state: weather_state
output_weather_notlocation:
  type: message_text
  properties:
    text: "I did not understand the location, can you please repeat?"
  transitions:
    next_state: input_intent
check_jarvis_error:
  type: conditional_exists
  properties:
    key: '{{jarvis_error}}'
  transitions:
    exists: show_jarvis_api_error
    notexists: output_weather_notlocation
show_jarvis_api_error:
  type: message_text
  properties:

```

```
    text: "I am having troubled understanding right now. Come again on
that, else check jarvis services?"
    transitions:
      next_state: input_intent
```

## error\_flow.yml

```
name: error_flow
states:
  error_state:
    type: message_text_random
    properties:
      responses:
        - "Sorry I didn't get that!"
        - "Are we talking about retail or weather? What would you like to
know?"
        - "Sorry I know only about retail information or the weather"
        - "You can ask me about retail information or the weather, the
rainfall, the temperature. I don't know much more"
        - "Let's talk about retail or the weather!"
      delay: 0
    transitions:
      next_state: input_intent
```

[Next: Connect to Third-Party APIs as Fulfillment Engine](#)

## Copyright Information

Copyright © 2021 NetApp, Inc. All rights reserved. Printed in the U.S. No part of this document covered by copyright may be reproduced in any form or by any means-graphic, electronic, or mechanical, including photocopying, recording, taping, or storage in an electronic retrieval system-without prior written permission of the copyright owner.

Software derived from copyrighted NetApp material is subject to the following license and disclaimer:

THIS SOFTWARE IS PROVIDED BY NETAPP "AS IS" AND WITHOUT ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE, WHICH ARE HEREBY DISCLAIMED. IN NO EVENT SHALL NETAPP BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

NetApp reserves the right to change any products described herein at any time, and without notice. NetApp assumes no responsibility or liability arising from the use of products described herein, except as expressly agreed to in writing by NetApp. The use or purchase of this product does not convey a license under any patent rights, trademark rights, or any other intellectual property rights of NetApp.

The product described in this manual may be protected by one or more U.S. patents, foreign patents, or pending applications.

RESTRICTED RIGHTS LEGEND: Use, duplication, or disclosure by the government is subject to restrictions as set forth in subparagraph (c)(1)(ii) of the Rights in Technical Data and Computer Software clause at DFARS 252.277-7103 (October 1988) and FAR 52-227-19 (June 1987).

## Trademark Information

NETAPP, the NETAPP logo, and the marks listed at <http://www.netapp.com/TM> are trademarks of NetApp, Inc. Other company and product names may be trademarks of their respective owners.