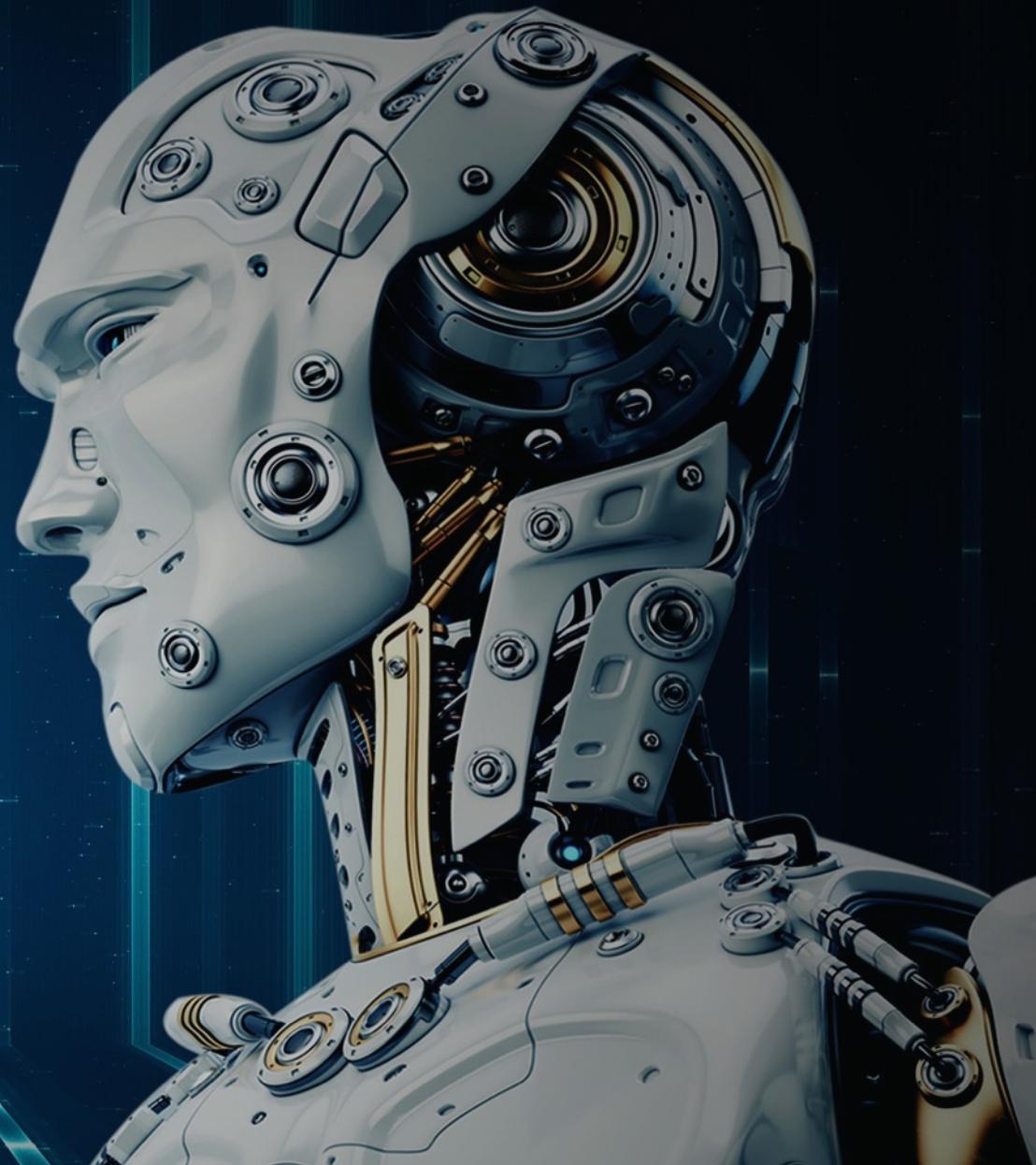


**ORACLE®**

# Oracle Intelligent Bots Advanced Training

Composite Bag Entities



# Safe Harbor Statement

The following is intended to outline our general product direction. It is intended for information purposes only, and may not be incorporated into any contract. It is not a commitment to deliver any material, code, or functionality, and should not be relied upon in making purchasing decisions. The development, release, and timing of any features or functionality described for Oracle's products remains at the sole discretion of Oracle.

# Topic Agenda

- 1 ➤ About Entities
- 2 ➤ Entity Slotting
- 3 ➤ Introduction to Composite Bag Entities
- 4 ➤ Example: Creating an Composite Bag Entity
- 5 ➤ Comparison With Other Entities

# Composite Bag Entities

## About Entities



Entity

- Variable/parameter for intent
- Important word in an input
- Adds relevance to intent
- Possibly maps to domain object

Check Balance  
Entity: AccountType

- Checking
- Savings
- Credit Card

“How much money do I have in my checking account?”

Entity

Variable/parameter for intent

Important word in an input

Adds relevance to intent

Possibly maps to domain object

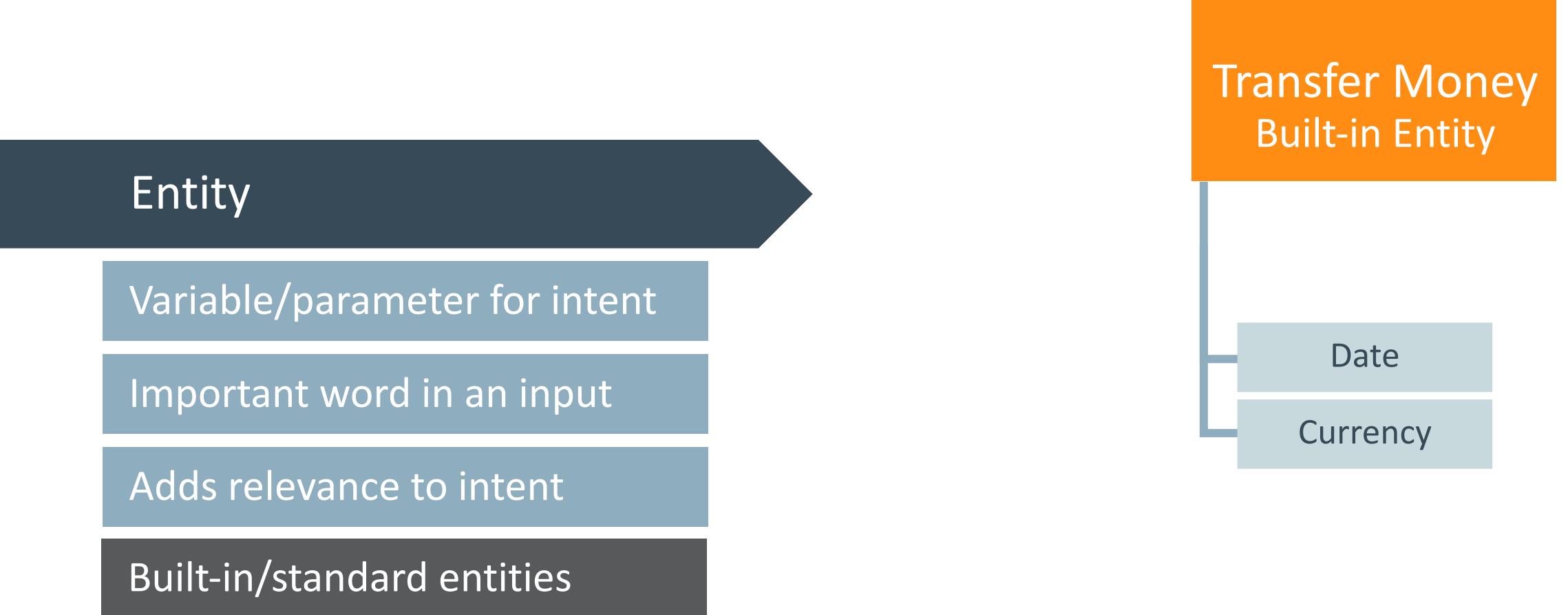
Check Balance  
Entity: AccountType

Checking

Savings

Credit Card

“How much money do I have in my **checking** account?”



Entity

Variable/parameter for intent

Important word in an input

Adds relevance to intent

Built-in/standard entities

Transfer Money  
Built-in Entity

Date

Currency

```
graph LR; Entity[Entity] --> Var[Variable/parameter for intent]; Entity --> Word[Important word in an input]; Entity --> Relevance[Adds relevance to intent]; Entity --> BuiltIn[Built-in/standard entities]; TransferMoney[Transfer Money  
Built-in Entity] --- Date[Date]; TransferMoney --- Currency[Currency]
```

Entity

Variable/parameter for intent

Important word in an input

Adds relevance to intent

Built-in/standard entities

Transfer Money  
Built-in Entity

Date

Currency

“Transfer \$50 to savings tomorrow”

Entity

Variable/parameter for intent

Important word in an input

Adds relevance to intent

Built-in/standard entities

Transfer Money  
Built-in Entity

Date

Currency

“Transfer \$50 to savings tomorrow”

# Composite Bag Entities

## Entity Slotting

# Entity slotting

```
startBalances:  
  component: "System.SetVariable"  
  properties:  
    variable: "accountType"  
    value: "${iResult.value.entityMatches['AccountType'][0]}"  
  transitions: {}  
  
askBalancesAccountType:  
  component: "System.List"  
  properties:  
    options: "${accountType.type.enumValues}"  
    prompt: "For which account do you want your balance?"  
    variable: "accountType"  
  transitions: {}  
  
-----  
  
| askBalancesAccountType:  
|   component: "System.List"  
|   properties:  
|     options: "${accountType.type.enumValues}"  
|     nlpResultVariable: "iResult"  
|     prompt: "For which account do you want your balance?"  
|     variable: "accountType"  
|   transitions: {}
```

- User input components are skipped if the variable they reference has a value set
  - Usually value is set from System.Intent
- Two approaches
  - Use Combination of System.SetVariable and input component
  - Use component nlpResultVariable property

# Discussion

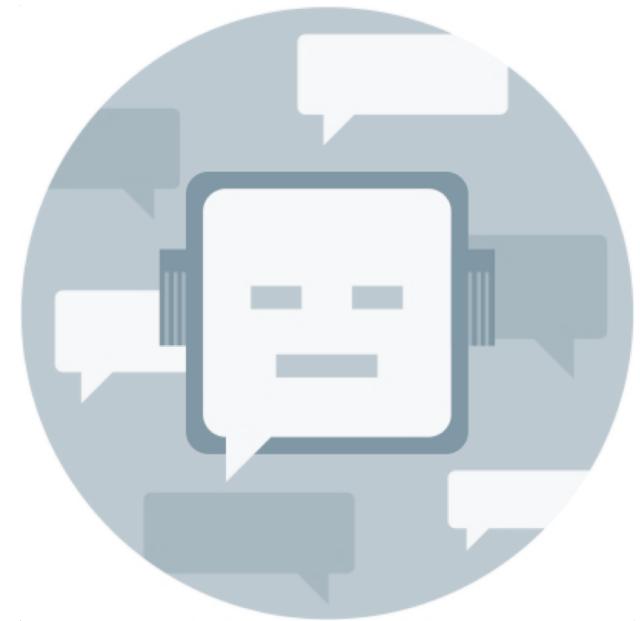
**When would you use which approach?**



# Composite Bag Entities

## Introduction

**A composite bag entity is a grouping of related entities that illustrates a larger concept like a business domain.**



# Composite Bag Entity

- Models a business domain object
- Composite bag entities are comprised of one to many entities
  - Custom entities
  - Built-in entities
- All contained entities get resolved in a single dialog flow state
  - System.ResolveEntities
  - System.CommonResponse
- Use sensible (logical) order when adding entities
  - Determines the sequence in which users are asked to provide missing entity values

# Order Pizza Dialog Flow With Composite Bag Entity

variables:  
order: "PizzaOrder" ←  
iResult: "nlpresult"

Configure

states:  
intent:  
component: "System.Intent" ← "A **cheese** pizza please."  
properties:  
variable: "iResult"

orderState:  
component: "System.ResolveEntities"  
properties:  
variable: "order"  
nlpResultVariable: "iResult"  
[...]

Composite Bag:  
PizzaOrder

Entity: PizzaType

Entity: PizzaSize

Entity: PizzaCrust

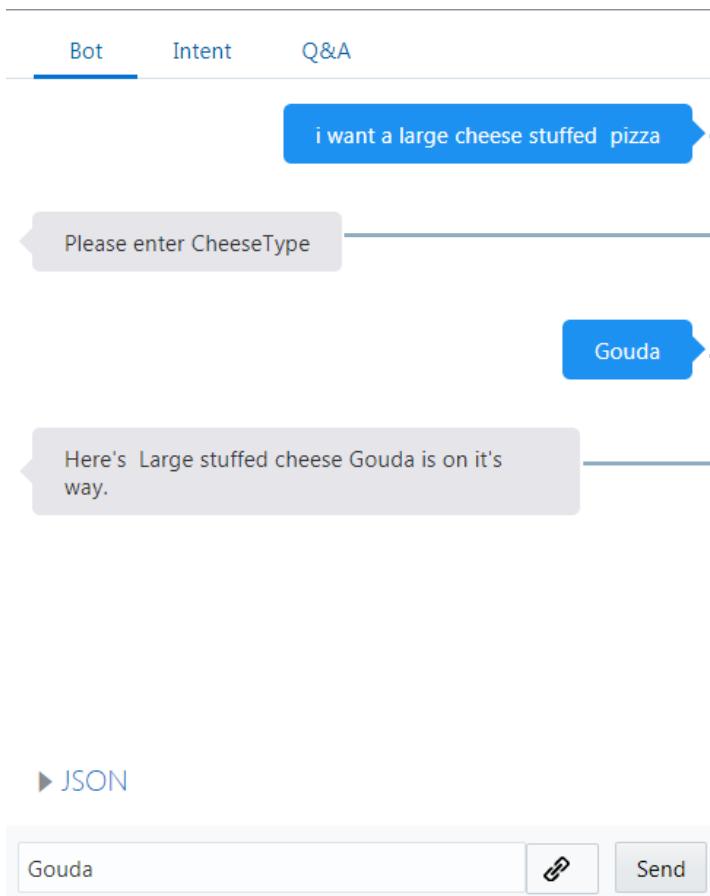
Check for Resolved Entities

Which size do you  
like for your pizza?

Which crust do you  
like?

PizzaType  
 PizzaSize  
 PizzaCrust

# Example



i want a large cheese stuffed pizza → OrderPizza intent having PizzaBag entity associated with it is resolved

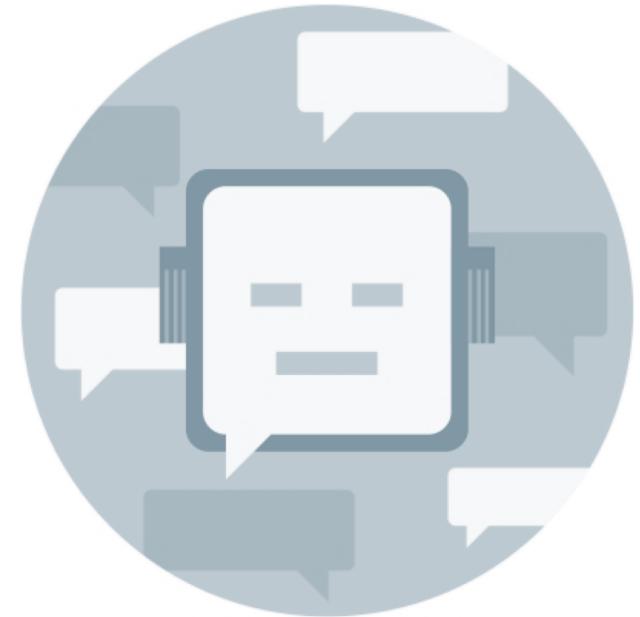
Please enter CheeseType → Prompt is invoked since CheeseType entity information was missing

Gouda → User enters the Gouda as CheeseType value

Here's Large stuffed cheese Gouda is on its way. → At this point all the listed entities in composite bag are resolved!

► JSON

The order in which entity values are provided in a **user input** sentence **does not need to match the order** in which entities are defined **in the composite bag entity**



# Composite Bag Entities

**Creating a Composite Bag Entity**

# Add One or Multiple Prompts to Custom Entities

Filter

Sort By Type Ascending

PizzaBag	X
CheeseType	X
PizzaCrust	X
PizzaSize	X
PizzaType	X
ADDRESS	X
CURRENCY	X
DATE	X
DURATION	X
EMAIL	X
NUMBER	X
PHONE_NUMBER	X
SET	X

Name \*

Description

Configuration

Type ?

+ Value

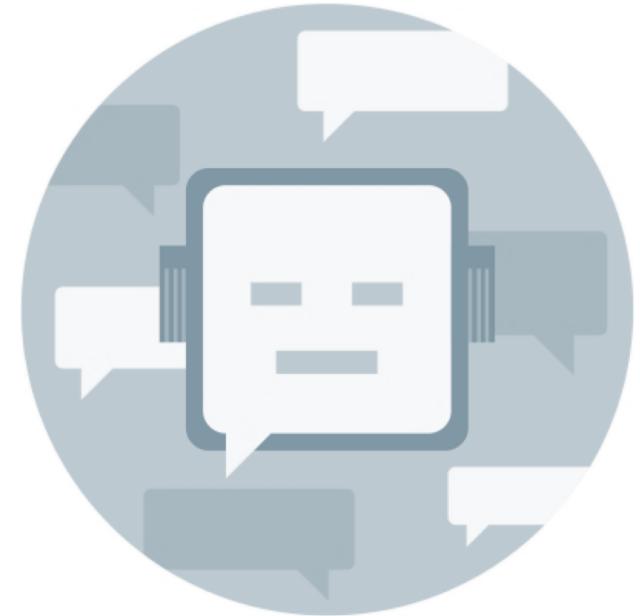
Value	Synonyms
Mozzarella	Mozarela, Mozzarella
Gouda	

+ Prompt

Prompt ?

Please enter CheeseType

**If no prompt is defined by the entity designer then a default prompts gets generated**



# Create an Entity of Type "Composite Bag"

The screenshot shows the Oracle Database Entity Creation Wizard interface. On the left is a vertical toolbar with icons for Home, Entity (+), More, Filter, Sort By, Description, Configuration, Composite Entities, and Help.

**Description**

- Name \*: PizzaBag
- Description: (empty)

**Configuration**

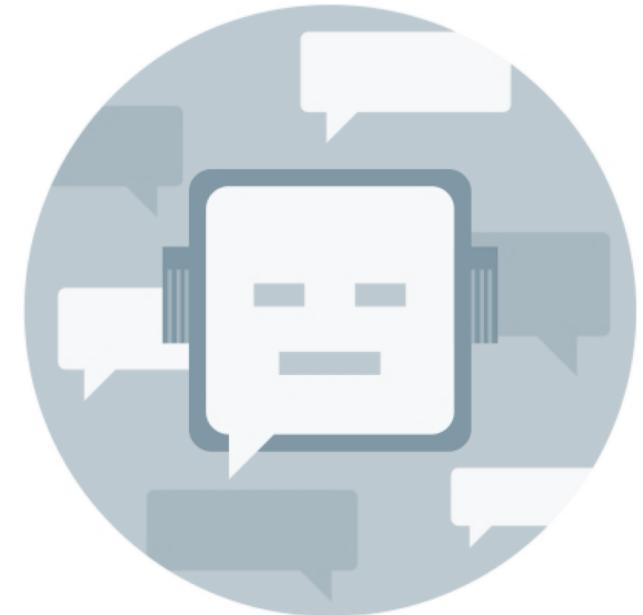
- Type: Composite Bag

**Composite Entities**

- CheeseType
- PizzaCrust
- PizzaSize
- PizzaType

The "Type" field and the "Composite Entities" list are highlighted with red boxes.

Composite bag entities **cannot contain**  
other composite bag entities



# Associate Composite Bag Entity with an Intent

The screenshot shows two panels from the Oracle Service Cloud interface. The left panel displays an intent named 'OrderPizza'. The right panel shows the 'Intent Entities' section where a composite bag entity named 'PizzaBag' is listed. A red box highlights the 'PizzaBag' entry.

+ Intent More ▾

Description

Name \* OrderPizza

Page 1 of 1 (1 of 1 items) Description

Intent Entities + Entity ▾

PizzaBag

# Add Composite Bag Entity to Dialog Flow

```
5 context:  
6   variables:  
7     pizza: "PizzaBag"  
8     iResult: "nlpresult"  
9 states:  
10   intent:  
11     component: "System.Intent"  
12     properties:  
13       variable: "iResult"  
14       confidenceThreshold: 0.4  
15   transitions:  
16     actions:  
17       OrderPizza: "resolvepizza"  
18       unresolvedIntent: "unresolved"  
19   resolvepizza:  
20     component: "System.ResolveEntities"  
21     properties:  
22       variable: "pizza"  
23       nlpResultVariable: "iResult"  
24     transitions: {}  
25 done:  
26   component: "System.Output"  
27   properties:  
28     text: "Here's ${pizza.value.PizzaSize[0]} ${pizza.value.PizzaCrust[0]} ${pizza.value.PizzaType[0]} ${pizza.value.CheeseType[0]} is on it's way."  
29   transitions:  
30     return: "done"  
31 unresolved:  
32   component: "System.Output"  
33   properties:  
34     text: "I don't understand. What do you want to do?"  
35   transitions:  
36     return: "unresolved"
```

# System.ResolveEntities Component Properties

The System.ResolveEntities built-in component uses the prompts defined on the custom entities to request input from the user

variable	The name of the context or user variable
variable	References a context variable of a composite bag type
nlpResultVariable	Refers to the nlpresult variable that can be used to resolve the composite entity variable.
maxPrompts	(Optional) specifies the number of attempts allotted to the user to enter a valid value that matches the child entity type.
entityOrder	A comma-delimited list of entity names. It specifies the order in which entities in the composite bag will be resolved. If you omit one or more entities that are in the bag, they will be resolved at the end.
translate	Allows you to override the global autoTranslate variable

# Composite Bag Entities

**Comparison With Other Entities**

# Context Variables

## Individual Entities

- Declare context variable for **several** entities:

```
context:  
variables:  
size: "PizzaSize"  
type: "PizzaType"  
crust: "PizzaCrust"  
iResult: "nlpresult"
```

## Composite Bag Entity

- Declare the **composite bag** entity:

```
context:  
variables:  
pizza: "PizzaBag"  
iResult: "nlpresult"
```

# Setting Values

## Individual Entities

- Add values to individual context variables
  - System.SetVariable
  - System.MatchEntity
  - System.Text
  - System.List
  - System.CommonResponse

## Composite Bag Entity

- Request user input
  - System.ResolveEntities
  - System.CommonResponse

```
resolvePizza:  
  component: "System.ResolveEntities"  
  properties:  
    variable: "pizza"  
    nlpResultVariable: "iResult"  
    entityOrder: "PizzaSize,PizzaType,CheeseType,PizzaCrust"
```

# Number of States Required

## Individual Entities

- Require a series of states to collect missing information from the user

## Composite Bag Entity

- A single state resolves all missing entity values
  - System.ResolveEntities
  - System.CommonResponse

# Referencing Entity Values

## Individual Entities

- Use  `${variableName.value}`  expressions

```
done:  
  component: "System.Output"  
  properties:  
    text: "Your ${size.value} ${type.value} Pizza is on its way."  
  transitions:  
    return: "done"
```

## Composite Bag Entity

- Reference the fields of the entity  
 `${variableName.value.entityName[0]}` :

```
done:  
  component: "System.Output"  
  properties:  
    text: "Here's ${pizza.value.PizzaSize[0]} ${pizza.value.PizzaCrust[0]}  
${pizza.value.PizzaType[0]} ${pizza.value.CheeseType[0]} is on it's way."  
  transitions:  
    return: "done"
```

# Integrated Cloud Applications & Platform Services

**ORACLE®**