Calculator Component

This component will help perform basic arithmetic calculations calculations

It has 2 property

- **inputStringNumbers** - this will receive a set of numbers such as 1,3,10,12 or 30,20 .

The number set has to be as a string e.g: “30,20”

- **operation** - It can have one of this values:

addition

subtraction

division

multiplication

arithmeticAverage

Example

```
main: true
name: test_this
context:
 # Define the variables which will be used throughout the dialog flow here.
  # The syntax for defining the variables is variablename: "variableType".
  # The "variableType" can be defined as a primitive type ("string", "boolean", "int", "float", "double"), "list", "map" or an entity name.
  # The variable 'rb' of type 'resourcebundle' is pre-defined, and can be used to resolve resource bundle entries in the language of the us
  variables:
    # greeting is a variable that references the Greeting composite bag entity.
    # This composite bag entity has items to prompt for greeting type and name
    greeting: "Greeting"
    iResult: "nlpresult"

    my_num: "string"
# states is where you can define the various states within your flow.
# The syntax for defining a state is
# statename:
#   component:Specify the component you want to use. This can be either a built-in or custom component.
#   properties:
#     property1: "value" (These are the properties to the specified component
#   transitions: You can specify one of the following four
#     next: Specify the state you want to execute next after this state. By default it will execute the state specified after this
#     error: Specify the state you want to execute in case the component encounters any error in execution.
#     actions: You can handle various actions returned by the components here the syntax is actionName: "statename"
#        action1: state1
#     return: "done" You can exit the flow using the return statement
states:

  intent:
    component: "System.Intent"
    properties:
      variable: "iResult"
    transitions:
      actions:
      next: "setVariable"


  setVariable:
    component: "System.SetVariable"
    properties:
      # variable refers to the context or user variable that gets the value specified in the value property.
      variable: "my_num"
      # value set for the variable.
      value: "6,3"
  print_t:
    component: "System.CommonResponse"
    properties:
      keepTurn: true
      # metadata property specifies the structure of the bot response message(s) that are sent to the user. See the documentation of the Sy
      metadata:
        responseItems:
          - type: "text"
            text: "${my_num}"
    transitions:
      next: "calculate"


  calculate:
    # change the name of the component to your custom component name
    component: "calculator"
    properties:
      # specify the properties the custom component supports
      inputStringNumbers: "${my_num}"
      operation: "division"
    transitions:
      next: "print_tt"

  print_tt:
    component: "System.CommonResponse"
    properties:
      # metadata property specifies the structure of the bot response message(s) that are sent to the user. See the documentation of the Sy
      metadata:
        responseItems:
          - type: "text"
            text: "${user.calculationResult}"
    transitions:
      return: "done"
```