本地 python 技能调试环境搭建说明-Fiberhome版本

python安装：

windows

- 1.1 下载官网python3.6.8 解压免安装版本。

下载地址：https://www.python.org/ftp/python/3.6.8/python-3.6.8-embed-amd64.zip

- 1.2 解压后，添加路径 环境变量=》系统变量PATH 最前面

例如：D:\installsoftware\developer\Python\python-3.6.8\

- 1.3 安装pip工具
  - 删除python安装目录中的python36._pth
  - 通过https://bootstrap.pypa.io/get-pip.py下载pip脚本(项目根目录中已经存在了)
  - get-pip.py所在目录执行命令"python get-pip.py"
  - 添加路径 环境变量=》系统变量PATH: "D:\installsoftware\developer\Python\python-3.6.8\Scripts\"

验证是否按照python成功

执行如下两条命令：

```
$ python -V
Python 3.6.10

$ pip -V
pip 18.1 from /opt/python3.6/lib/python3.6/site-packages/pip (python 3.6)
```

若得到类似输出，则代表安装成功。**特别注意pip版本、路径与前面步骤安装的版本一致。**

获取azero本地调试项目并搭建环境

获取版本

```
https://github.com/sai-azero/azero-skills-kit-sdk-for-python
```

安装项目根目录下lib中的所有安装包

(在lib目录下执行pip install xxx.tar.gz)，例如:

```
pip install azero-sdk-2.0.2.tar.gz
pip install azero-sdk-util-1.0.3.tar.gz
pip install azero-sdk-mongodb-persistence-adapter-1.0.1.tar.gz
```

安装必要库：

```
pip install -r requirements.txt
```

注意：如果某个库无法下载，请删除requirements.txt中的库的版本号。

运行python SDK并使用postman测试

运行python SDK

执行如下命令。

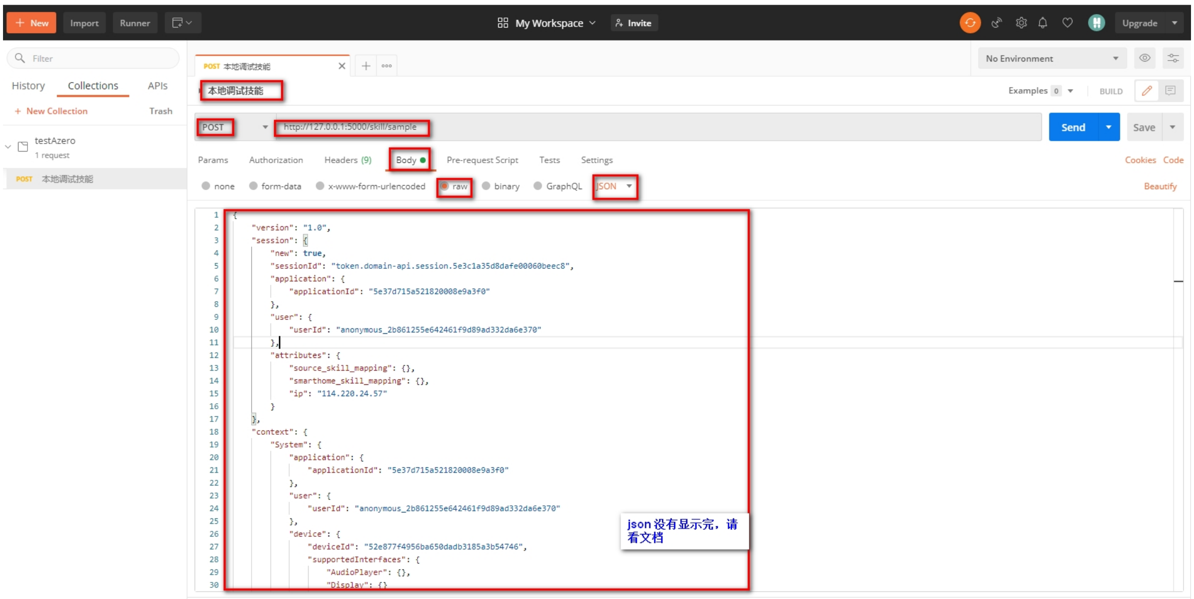
```
$ python app.py
 * Serving Flask app "app" (lazy loading)
 * Environment: production
   WARNING: This is a development server. Do not use it in a production
deployment.
   Use a production WSGI server instead.
 * Debug mode: off
INFO:werkzeug: * Running on http://127.0.0.1:5000/ (Press CTRL+C to quit)
```

若得到同样输出，或使用浏览器访问所提示的网址，看到"启动成功"字样，则代表运行成功。

使用postman测试

保持python SDK运行，使用postman使用POST方法发送如下json串到地址http://127.0.0.1:5000/skill/sample *注：skill/sample与index.py所在路径保持一致*

```
{
    "version": "1.0",
    "session": {
        "new": true,
        "sessionId": "token.domain-api.session.5e3c1a35d8dafe00060beec8",
        "application": {
            "applicationId": "5e37d715a521820008e9a3f0"
        },
        "user": {
            "userId": "anonymous_2b861255e642461f9d89ad332da6e370"
        },
        "attributes": {
            "source_skill_mapping": {},
            "smarthome_skill_mapping": {},
            "ip": "114.220.24.57"
        }
    },
    "context": {
        "System": {
            "application": {
                "applicationId": "5e37d715a521820008e9a3f0"
            },
            "user": {
                "userId": "anonymous_2b861255e642461f9d89ad332da6e370"
            },
            "device": {
                "deviceId": "52e877f4956ba650dadb3185a3b54746",
                "supportedInterfaces": {
                    "AudioPlayer": {},
                    "Display": {}
                }
            }
        }
    },
    "request": {
        "type": "IntentRequest",
        "requestId": "e4308cca-6e8f-488b-a365-da70453972cf",
        "timestamp": "2020-02-14T00:52:53.541Z",
        "dialogState": "COMPLETED",
        "intent": {"如果需要命中意图并获取意图标识以及槽位信息在这里处理"}
    }
}
```



POST 本地调试技能

本地调试技能

POST http://127.0.0.1:5000/skill/sample

Body ●

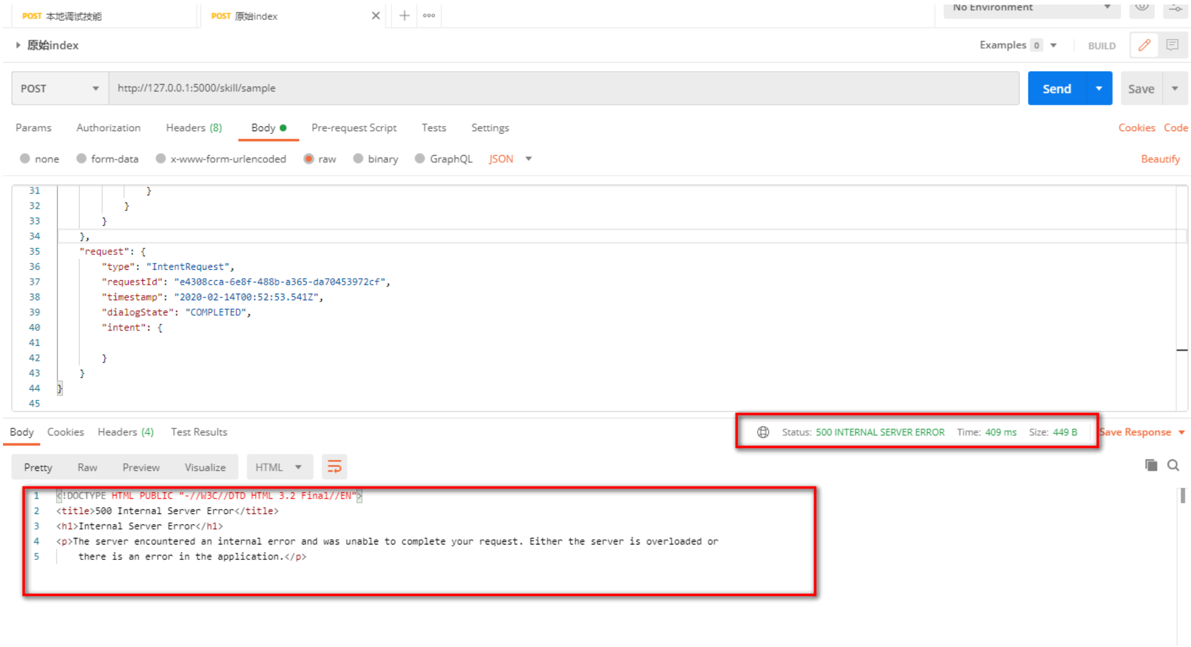
○ none ○ form-data ○ x-www-form-urlencoded ● raw ○ binary ○ GraphQL JSON

```
{
    "version": "1.0",
    "session": {
        "new": true,
        "sessionId": "token.domain-api.session.5e3c1a35d8dafe00060beec8",
        "application": {
            "applicationId": "5e37d715a521820008e9a3f0"
        },
        "user": {
            "userId": "anonymous_2b861255e642461f9d89ad332da6e370"
        },
        "attributes": {
            "source_skill_mapping": {},
            "smarthome_skill_mapping": {},
            "ip": "114.220.24.57"
        }
    },
    "context": {
        "System": {
            "application": {
                "applicationId": "5e37d715a521820008e9a3f0"
            },
            "user": {
                "userId": "anonymous_2b861255e642461f9d89ad332da6e370"
            },
            "device": {
                "deviceId": "52e877f4956ba650dadb3185a3b54746",
                "supportedInterfaces": {
                    "AudioPlayer": {},
                    "Display": {}
```

json 没有显示完，请看文档

遇到问题

第一次使用postman 会报错



POST http://127.0.0.1:5000/skill/sample

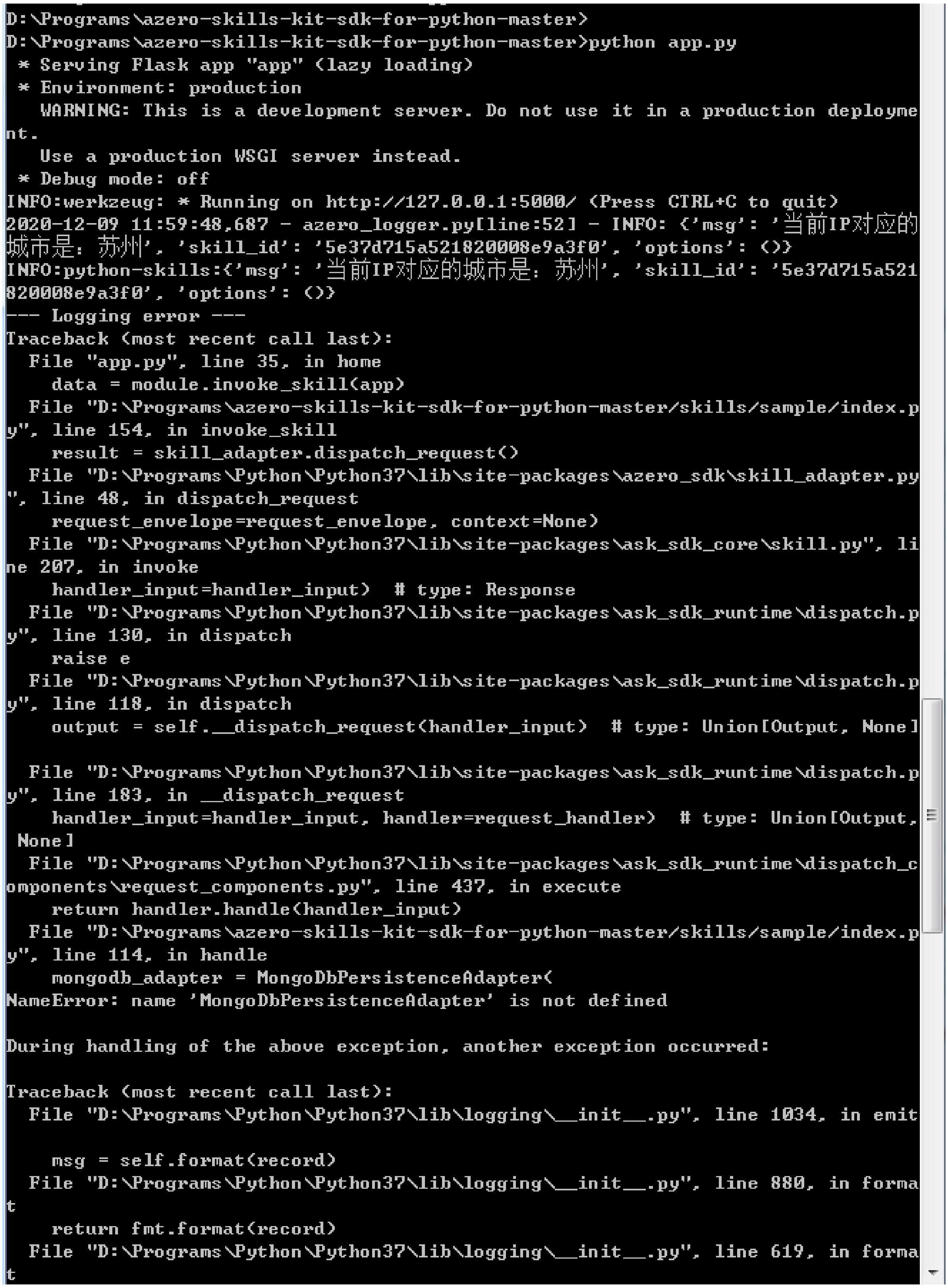
```
                }
            }
        }
    },
    "request": {
        "type": "IntentRequest",
        "requestId": "e4308cca-6e8f-488b-a365-da70453972cf",
        "timestamp": "2020-02-14T00:52:53.541Z",
        "dialogState": "COMPLETED",
        "intent": {

        }
    }
}
```

Status: 500 INTERNAL SERVER ERROR Time: 409 ms Size: 449 B

HTML

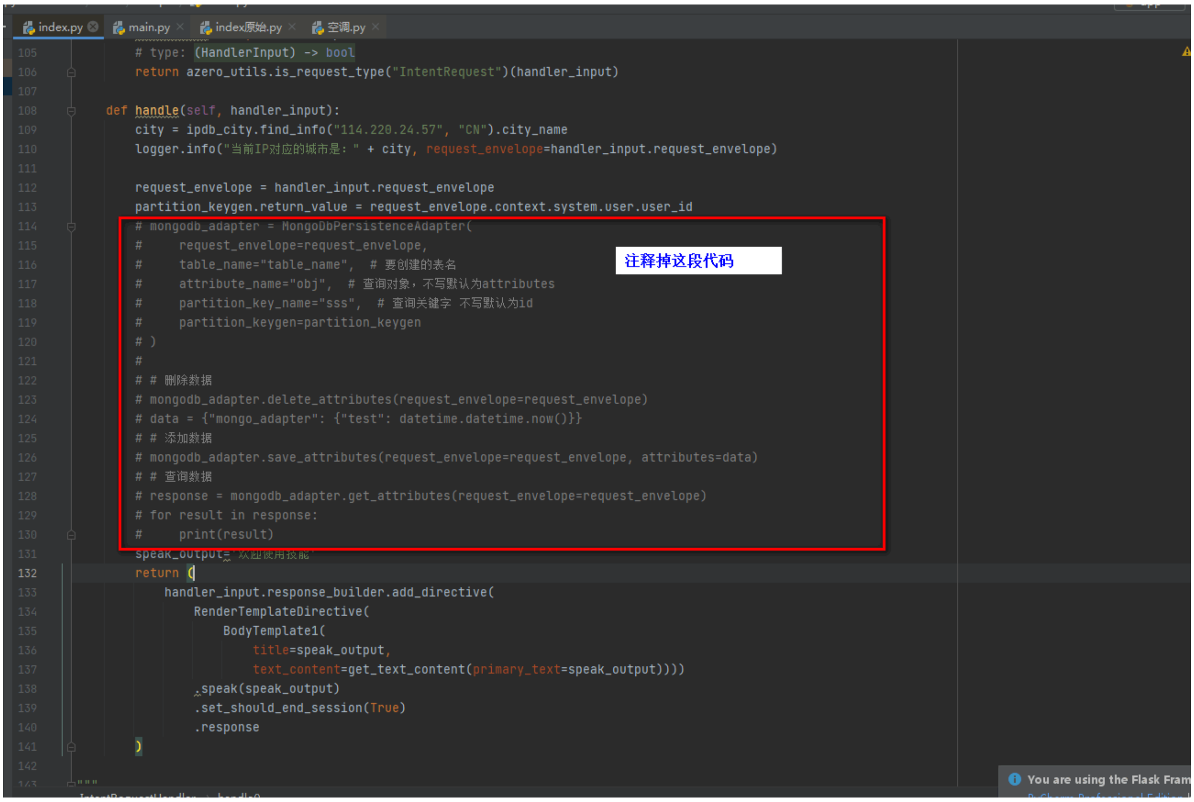
```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 3.2 Final//EN">
<title>500 Internal Server Error</title>
<h1>Internal Server Error</h1>
<p>The server encountered an internal error and was unable to complete your request. Either the server is overloaded or
    there is an error in the application.</p>
```

服务器报错



```
D:\Programs\azero-skills-kit-sdk-for-python-master>
D:\Programs\azero-skills-kit-sdk-for-python-master>python app.py
 * Serving Flask app "app" (lazy loading)
 * Environment: production
   WARNING: This is a development server. Do not use it in a production deployme
nt.
   Use a production WSGI server instead.
 * Debug mode: off
INFO:werkzeug: * Running on http://127.0.0.1:5000/ (Press CTRL+C to quit)
2020-12-09 11:59:48,687 - azero_logger.py[line:52] - INFO: {'msg': '当前IP对应的
城市是：苏州', 'skill_id': '5e37d715a521820008e9a3f0', 'options': {}}
INFO:python-skills:{'msg': '当前IP对应的城市是：苏州', 'skill_id': '5e37d715a521
820008e9a3f0', 'options': {}}
--- Logging error ---
Traceback (most recent call last):
  File "app.py", line 35, in home
    data = module.invoke_skill(app)
  File "D:\Programs\azero-skills-kit-sdk-for-python-master/skills/sample/index.p
y", line 154, in invoke_skill
    result = skill_adapter.dispatch_request()
  File "D:\Programs\Python\Python37\lib\site-packages\azero_sdk\skill_adapter.py
", line 48, in dispatch_request
    request_envelope=request_envelope, context=None)
  File "D:\Programs\Python\Python37\lib\site-packages\ask_sdk_core\skill.py", li
ne 207, in invoke
    handler_input=handler_input)  # type: Response
  File "D:\Programs\Python\Python37\lib\site-packages\ask_sdk_runtime\dispatch.p
y", line 130, in dispatch
    raise e
  File "D:\Programs\Python\Python37\lib\site-packages\ask_sdk_runtime\dispatch.p
y", line 118, in dispatch
    output = self.__dispatch_request(handler_input)  # type: Union[Output, None]

  File "D:\Programs\Python\Python37\lib\site-packages\ask_sdk_runtime\dispatch.p
y", line 183, in __dispatch_request
    handler_input=handler_input, handler=request_handler)  # type: Union[Output,
 None]
  File "D:\Programs\Python\Python37\lib\site-packages\ask_sdk_runtime\dispatch_c
omponents\request_components.py", line 437, in execute
    return handler.handle(handler_input)
  File "D:\Programs\azero-skills-kit-sdk-for-python-master/skills/sample/index.p
y", line 114, in handle
    mongodb_adapter = MongoDbPersistenceAdapter(
NameError: name 'MongoDbPersistenceAdapter' is not defined

During handling of the above exception, another exception occurred:

Traceback (most recent call last):
  File "D:\Programs\Python\Python37\lib\logging\__init__.py", line 1034, in emit

    msg = self.format(record)
  File "D:\Programs\Python\Python37\lib\logging\__init__.py", line 880, in forma
t
    return fmt.format(record)
  File "D:\Programs\Python\Python37\lib\logging\__init__.py", line 619, in forma
t
```

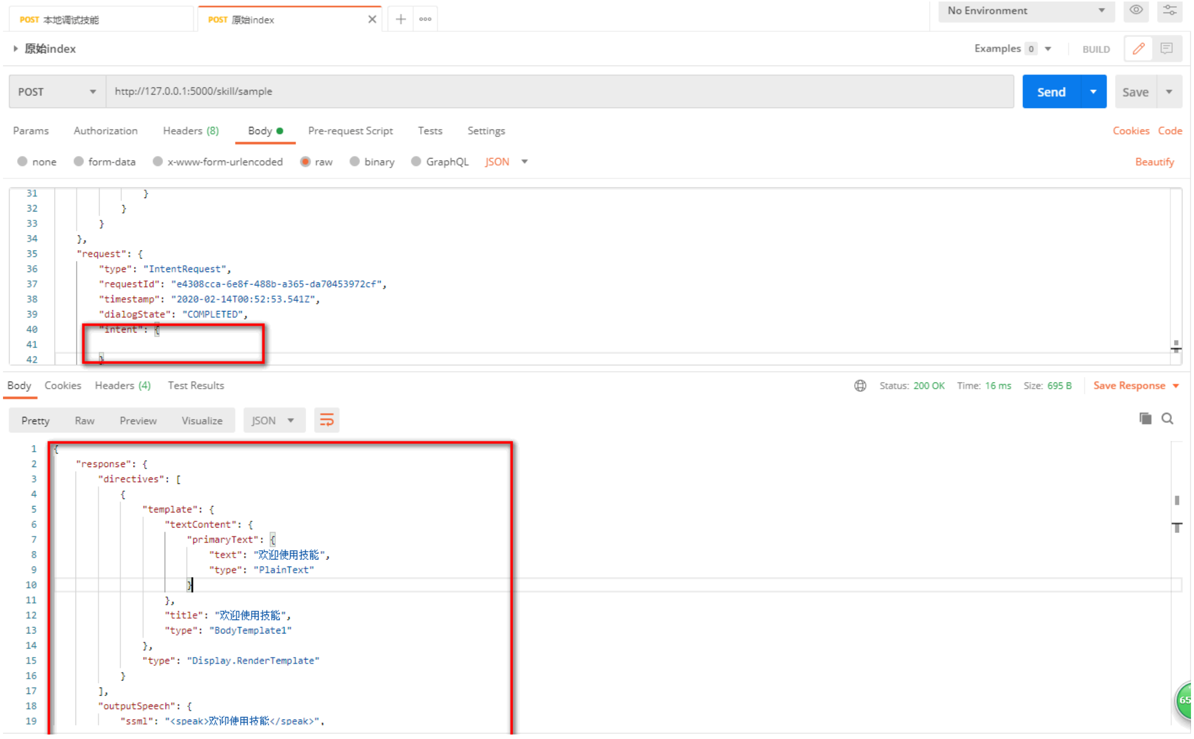
原因：在azero本地调试的项目工程中的index.py文件中的IntentRequestHandler 使用了Mongodb持久化的功能，将对于的代码注释掉即可。



```python
        # type: (HandlerInput) -> bool
        return azero_utils.is_request_type("IntentRequest")(handler_input)

    def handle(self, handler_input):
        city = ipdb_city.find_info("114.220.24.57", "CN").city_name
        logger.info("当前IP对应的城市是: " + city, request_envelope=handler_input.request_envelope)

        request_envelope = handler_input.request_envelope
        partition_keygen.return_value = request_envelope.context.system.user.user_id
        # mongodb_adapter = MongoDbPersistenceAdapter(
        #     request_envelope=request_envelope,
        #     table_name="table_name",  # 要创建的表名
        #     attribute_name="obj",  # 查询对象，不写默认为attributes
        #     partition_key_name="sss",  # 查询关键字 不写默认为id
        #     partition_keygen=partition_keygen
        # )
        #
        # # 删除数据
        # mongodb_adapter.delete_attributes(request_envelope=request_envelope)
        # data = {"mongo_adapter": {"test": datetime.datetime.now()}}
        # # 添加数据
        # mongodb_adapter.save_attributes(request_envelope=request_envelope, attributes=data)
        # # 查询数据
        # response = mongodb_adapter.get_attributes(request_envelope=request_envelope)
        # for result in response:
        #     print(result)
        speak_output='欢迎使用技能'
        return (
            handler_input.response_builder.add_directive(
                RenderTemplateDirective(
                    BodyTemplate1(
                        title=speak_output,
                        text_content=get_text_content(primary_text=speak_output)))
                .speak(speak_output)
                .set_should_end_session(True)
                .response
        )

"""
```

注释掉这段代码

注释代码后再postman试试



POST http://127.0.0.1:5000/skill/sample

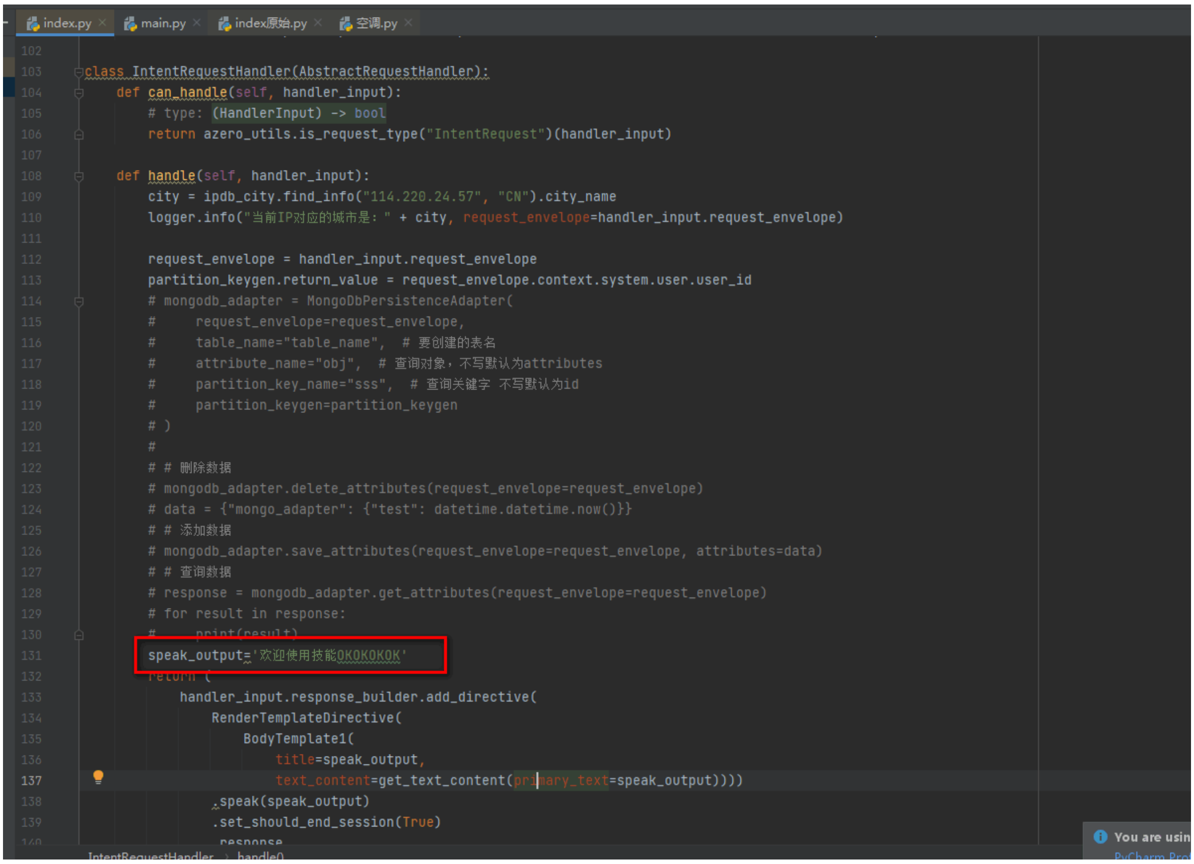
```json
                }
            }
        }
    },
    "request": {
        "type": "IntentRequest",
        "requestId": "e4308cca-6e8f-488b-a365-da70453972cf",
        "timestamp": "2020-02-14T00:52:53.541Z",
        "dialogState": "COMPLETED",
        "intent": {

        }
```

Status: 200 OK Time: 16 ms Size: 695 B

```json
{
    "response": {
        "directives": [
            {
                "template": {
                    "textContent": {
                        "primaryText": {
                            "text": "欢迎使用技能",
                            "type": "PlainText"
                        }
                    },
                    "title": "欢迎使用技能",
                    "type": "BodyTemplate1"
                },
                "type": "Display.RenderTemplate"
            }
        ],
        "outputSpeech": {
            "ssml": "<speak>欢迎使用技能</speak>",
```

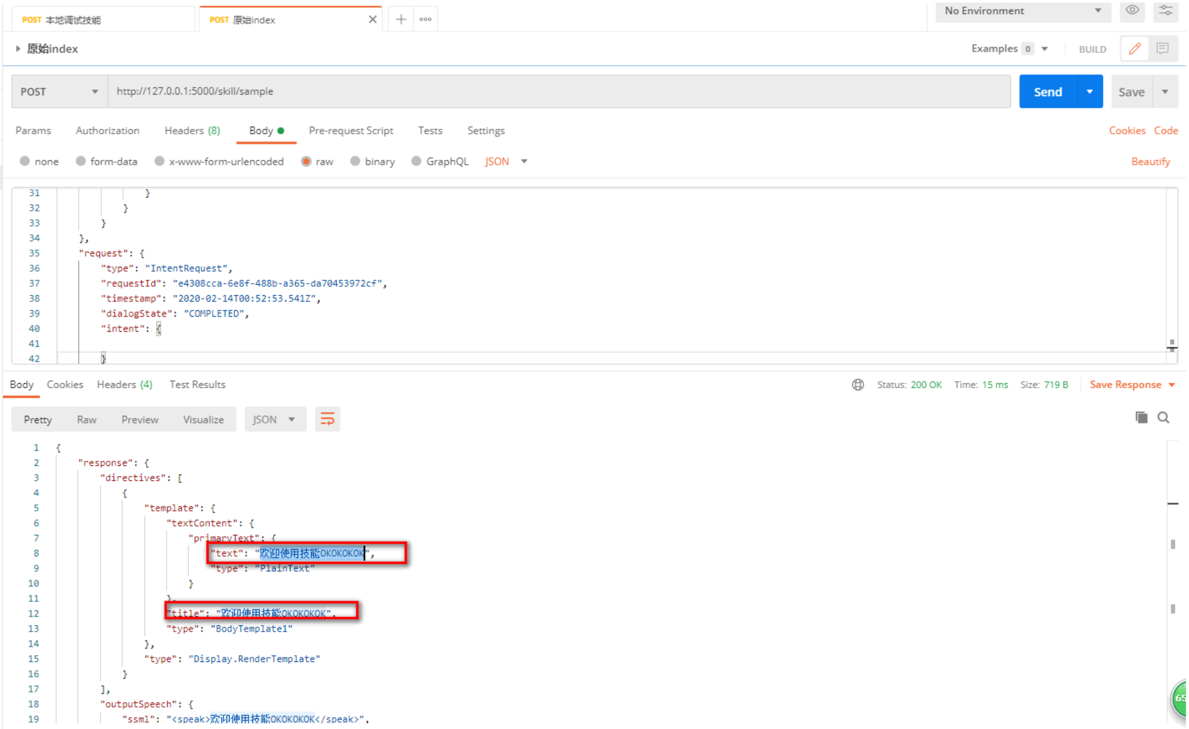
修改index中的代码后再试试，

修改如下：



```python
class IntentRequestHandler(AbstractRequestHandler):
    def can_handle(self, handler_input):
        # type: (HandlerInput) -> bool
        return azero_utils.is_request_type("IntentRequest")(handler_input)

    def handle(self, handler_input):
        city = ipdb_city.find_info("114.220.24.57", "CN").city_name
        logger.info("当前IP对应的城市是: " + city, request_envelope=handler_input.request_envelope)

        request_envelope = handler_input.request_envelope
        partition_keygen.return_value = request_envelope.context.system.user.user_id
        # mongodb_adapter = MongoDbPersistenceAdapter(
        #     request_envelope=request_envelope,
        #     table_name="table_name",  # 要创建的表名
        #     attribute_name="obj",  # 查询对象，不写默认为attributes
        #     partition_key_name="sss",  # 查询关键字 不写默认为id
        #     partition_keygen=partition_keygen
        # )
        #
        # # 删除数据
        # mongodb_adapter.delete_attributes(request_envelope=request_envelope)
        # data = {"mongo_adapter": {"test": datetime.datetime.now()}}
        # # 添加数据
        # mongodb_adapter.save_attributes(request_envelope=request_envelope, attributes=data)
        # # 查询数据
        # response = mongodb_adapter.get_attributes(request_envelope=request_envelope)
        # for result in response:
        #     print(result)
        speak_output='欢迎使用技能OKOKOKOK'
        return (
            handler_input.response_builder.add_directive(
                RenderTemplateDirective(
                    BodyTemplate1(
                        title=speak_output,
                        text_content=get_text_content(primary_text=speak_output)))
                .speak(speak_output)
                .set_should_end_session(True)
                .response
```

postman测试结果



POST http://127.0.0.1:5000/skill/sample

```
                }
            }
        }
    },
    "request": {
        "type": "IntentRequest",
        "requestId": "e4308cca-6e8f-488b-a365-da70453972cf",
        "timestamp": "2020-02-14T00:52:53.541Z",
        "dialogState": "COMPLETED",
        "intent": {

        }
```

Status: 200 OK Time: 15 ms Size: 719 B

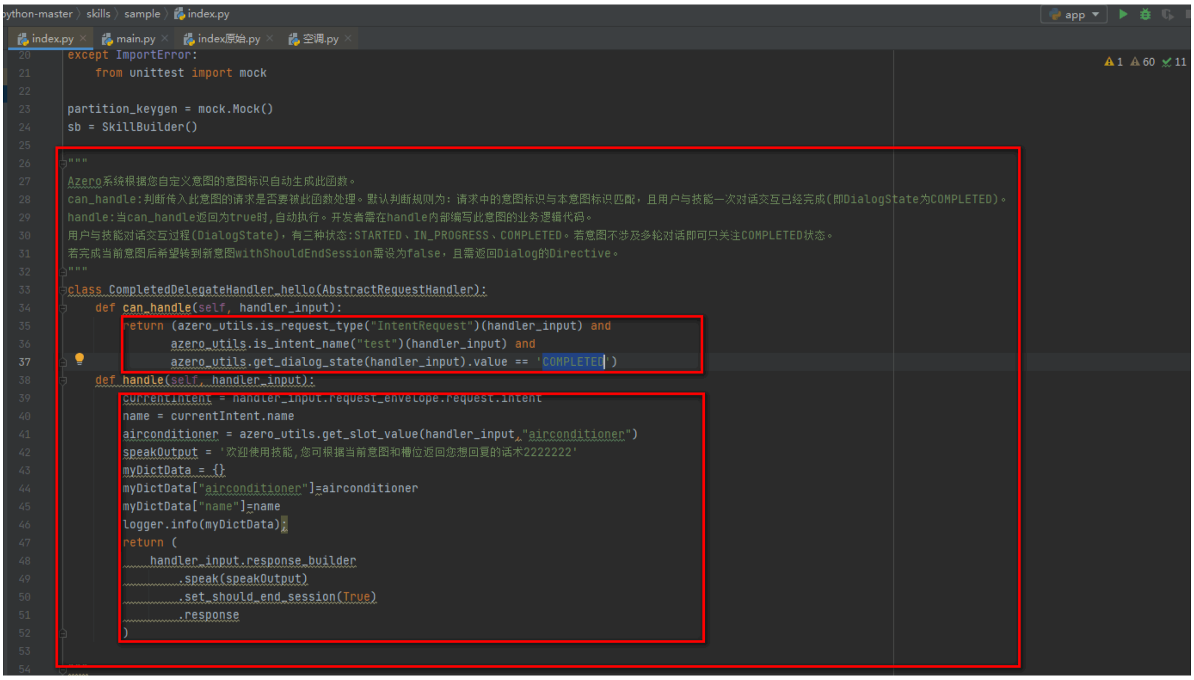
```
{
    "response": {
        "directives": [
            {
                "template": {
                    "textContent": {
                        "primaryText": {
                            "text": "欢迎使用技能OKOKOKOK",
                            "type": "PlainText"
                        }
                    },
                    "title": "欢迎使用技能OKOKOKOK",
                    "type": "BodyTemplate1"
                },
                "type": "Display.RenderTemplate"
            }
        ],
        "outputSpeech": {
            "ssml": "<speak>欢迎使用技能OKOKOKOK</speak>",
```

强化调试手段（重点）

上面的测试都只能走到每个技能的IntentRequestHandler 类中的handle方法中，那么怎么走到我们自定义的意图中呢？

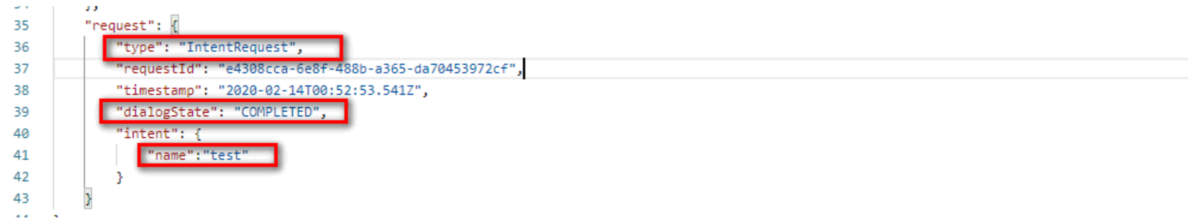
调试自定义的意图

比如：自定义意图



```python
except ImportError:
    from unittest import mock

partition_keygen = mock.Mock()
sb = SkillBuilder()

"""
Azero系统根据您自定义意图的意图标识自动生成此函数。
can_handle:判断传入此意图的请求是否要被此函数处理。默认判断规则为：请求中的意图标识与本意图标识匹配，且用户与技能一次对话交互已经完成(即DialogState为COMPLETED)。
handle:当can_handle返回为true时,自动执行。开发者需在handle内部编写此意图的业务逻辑代码。
用户与技能对话交互过程(DialogState)，有三种状态:STARTED、IN_PROGRESS、COMPLETED。若意图不涉及多轮对话即可只关注COMPLETED状态。
若完成当前意图后希望转到新意图withShouldEndSession需设为false，且需返回Dialog的Directive。
"""
class CompletedDelegateHandler_hello(AbstractRequestHandler):
    def can_handle(self, handler_input):
        return (azero_utils.is_request_type("IntentRequest")(handler_input) and
                azero_utils.is_intent_name("test")(handler_input) and
                azero_utils.get_dialog_state(handler_input).value == 'COMPLETED')
    def handle(self, handler_input):
        currentIntent = handler_input.request_envelope.request.intent
        name = currentIntent.name
        airconditioner = azero_utils.get_slot_value(handler_input,"airconditioner")
        speakOutput = '欢迎使用技能,您可根据当前意图和槽位返回您想回复的话术2222222'
        myDictData = {}
        myDictData["airconditioner"]=airconditioner
        myDictData["name"]=name
        logger.info(myDictData);
        return (
            handler_input.response_builder
                .speak(speakOutput)
                .set_should_end_session(True)
                .response
        )

"""
```

再看看postman 请求json数据中的request字段：



```json
    },
    "request": {
        "type": "IntentRequest",
        "requestId": "e4308cca-6e8f-488b-a365-da70453972cf",
        "timestamp": "2020-02-14T00:52:53.541Z",
        "dialogState": "COMPLETED",
        "intent": {
            "name":"test"
        }
    }
```

注意：intent中添加了 "name":"test"

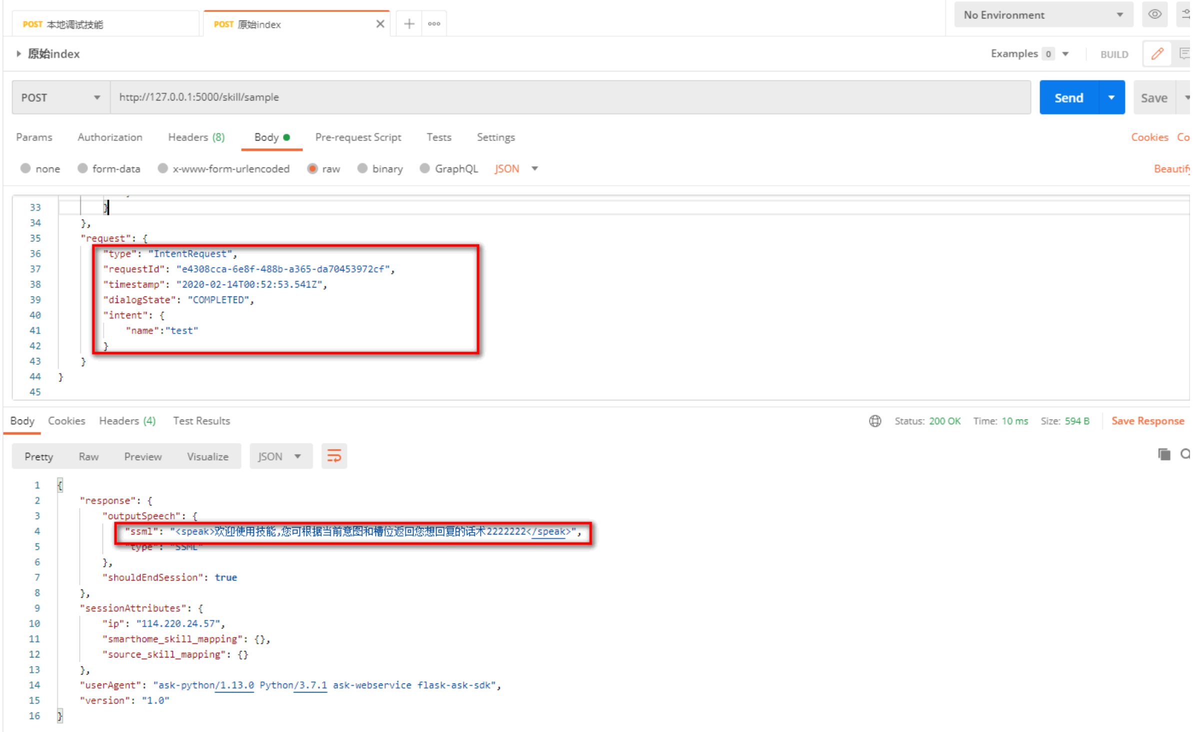
注意：

json中的type 就是python代码can_handle方法中的request_type

json中的dialogState就是python代码can_handle方法中的dialog_state

json中的name就是python代码can_handle方法中的intent_name

postman测试结果



POST http://127.0.0.1:5000/skill/sample

○ none ○ form-data ○ x-www-form-urlencoded ● raw ○ binary ○ GraphQL JSON

```
        }
    },
    "request": {
        "type": "IntentRequest",
        "requestId": "e4308cca-6e8f-488b-a365-da70453972cf",
        "timestamp": "2020-02-14T00:52:53.541Z",
        "dialogState": "COMPLETED",
        "intent": {
            "name":"test"
        }
    }
}
```

Body Cookies Headers (4) Test Results — Status: 200 OK Time: 10 ms Size: 594 B Save Response

```
{
    "response": {
        "outputSpeech": {
            "ssml": "<speak>欢迎使用技能,您可根据当前意图和槽位返回您想回复的话术2222222</speak>",
            "type": "SSML"
        },
        "shouldEndSession": true
    },
    "sessionAttributes": {
        "ip": "114.220.24.57",
        "smarthome_skill_mapping": {},
        "source_skill_mapping": {}
    },
    "userAgent": "ask-python/1.13.0 Python/3.7.1 ask-webservice flask-ask-sdk",
    "version": "1.0"
}
```

调试自定义意图标识和字典标识

疑问：

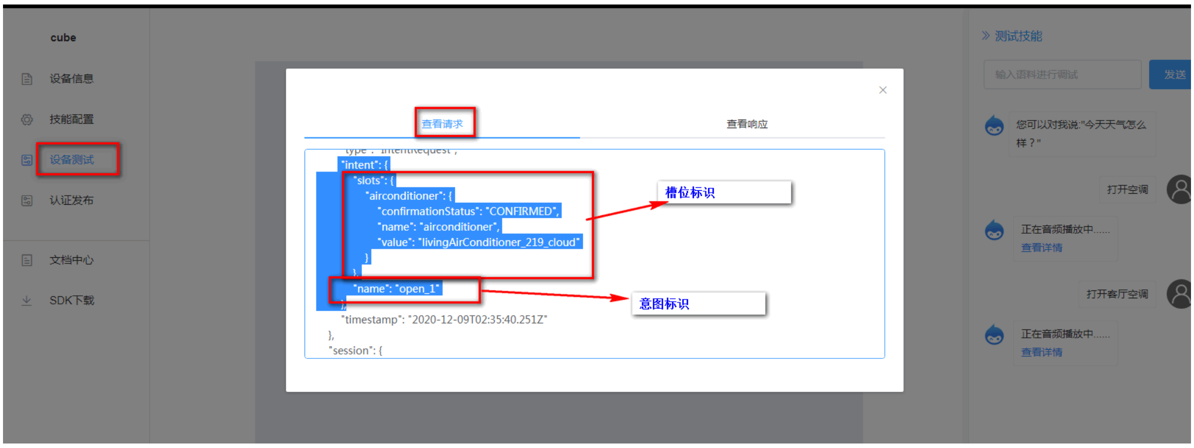
在python代码中需要获取意图标识和词典标识的时候，使用postman中的json数据如何配置呢？

1- 意图标识：

上面“调试自定义的意图” 中 postman json数据intent下新添加的 "name":"test" ， test就是意图标识；

2- 字典标识：

首先在azero平台测试中可以获取到槽位的格式



您可以对我说:"今天天气怎么样？"

打开空调

正在音频播放中......
查看详情

打开客厅空调

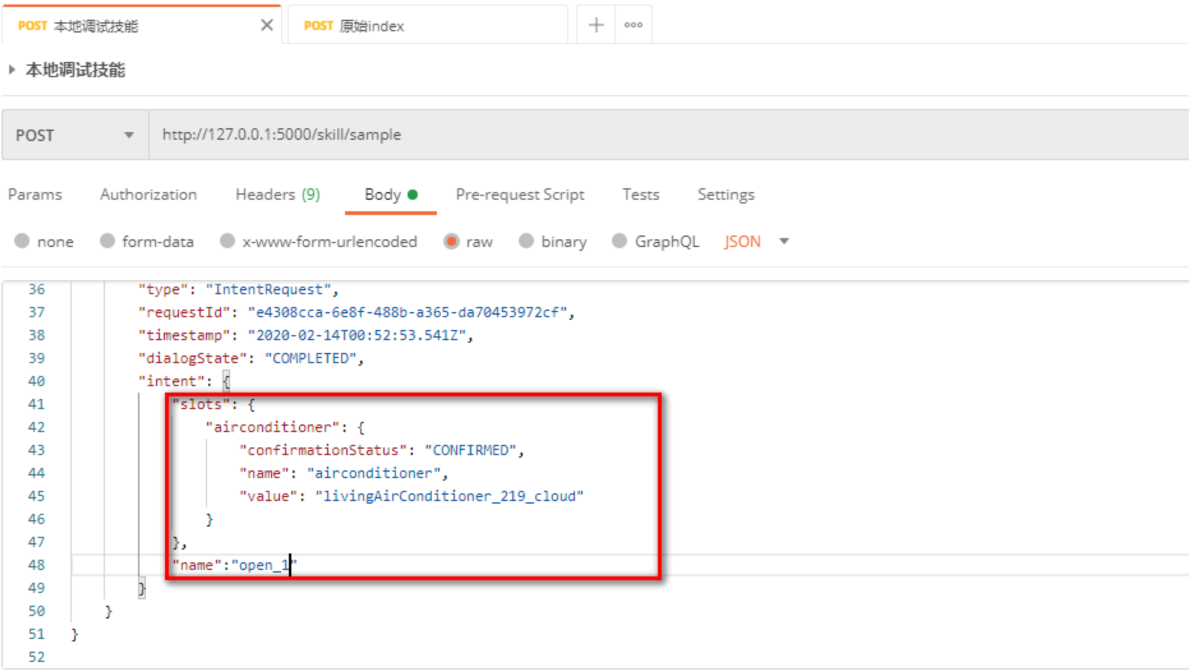
正在音频播放中......
查看详情

```
    "type": "IntentRequest",
    "intent": {
        "slots": {
            "airconditioner": {
                "confirmationStatus": "CONFIRMED",
                "name": "airconditioner",
                "value": "livingAirConditioner_219_cloud"
            }
        },
        "name": "open_1"
    },
    "timestamp": "2020-12-09T02:35:40.251Z"
},
"session": {
```

槽位信息的json格式为：

```
"intent": {
            "slots": {
                "airconditioner": {
                    "confirmationStatus": "CONFIRMED",
                    "name": "airconditioner",
                    "value": "livingAirConditioner_219_cloud"
                }
            },
            "name": "open_1"
        }
```

将slots数据放到postman请求的json格式的数据中

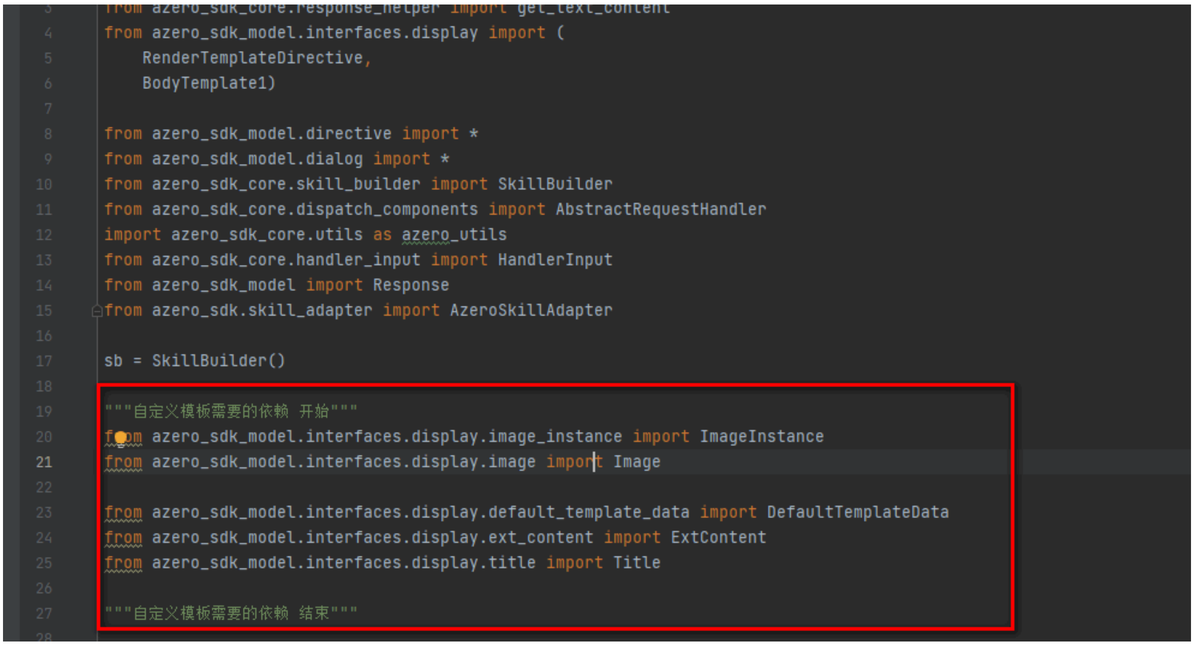


```
        "type": "IntentRequest",
        "requestId": "e4308cca-6e8f-488b-a365-da70453972cf",
        "timestamp": "2020-02-14T00:52:53.541Z",
        "dialogState": "COMPLETED",
        "intent": {
            "slots": {
                "airconditioner": {
                    "confirmationStatus": "CONFIRMED",
                    "name": "airconditioner",
                    "value": "livingAirConditioner_219_cloud"
                }
            },
            "name":"open_1"
        }
    }
}
```

python代码

1- 添加引用

```
"""自定义模板需要的依赖 开始"""
from azero_sdk_model.interfaces.display.image_instance import ImageInstance
from azero_sdk_model.interfaces.display.image import Image

from azero_sdk_model.interfaces.display.default_template_data import
DefaultTemplateData
from azero_sdk_model.interfaces.display.ext_content import ExtContent
from azero_sdk_model.interfaces.display.title import Title

"""自定义模板需要的依赖 结束"""
```

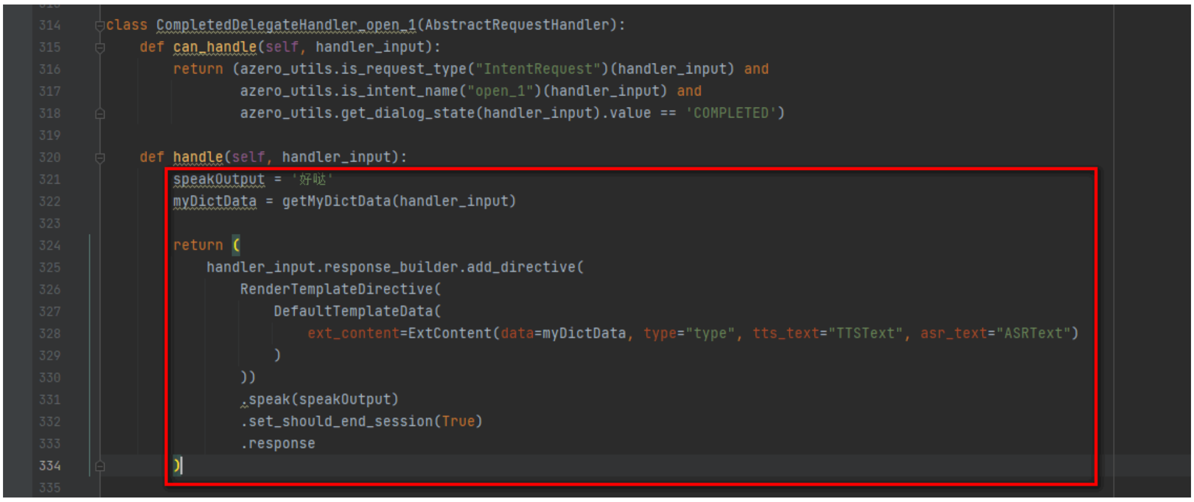


```python
from azero_sdk_core.response_helper import get_text_content
from azero_sdk_model.interfaces.display import (
    RenderTemplateDirective,
    BodyTemplate1)

from azero_sdk_model.directive import *
from azero_sdk_model.dialog import *
from azero_sdk_core.skill_builder import SkillBuilder
from azero_sdk_core.dispatch_components import AbstractRequestHandler
import azero_sdk_core.utils as azero_utils
from azero_sdk_core.handler_input import HandlerInput
from azero_sdk_model import Response
from azero_sdk.skill_adapter import AzeroSkillAdapter

sb = SkillBuilder()

"""自定义模板需要的依赖 开始"""
from azero_sdk_model.interfaces.display.image_instance import ImageInstance
from azero_sdk_model.interfaces.display.image import Image

from azero_sdk_model.interfaces.display.default_template_data import DefaultTemplateData
from azero_sdk_model.interfaces.display.ext_content import ExtContent
from azero_sdk_model.interfaces.display.title import Title

"""自定义模板需要的依赖 结束"""
```

2- 使用自定义意图返回模板

DefaulTemplateData()

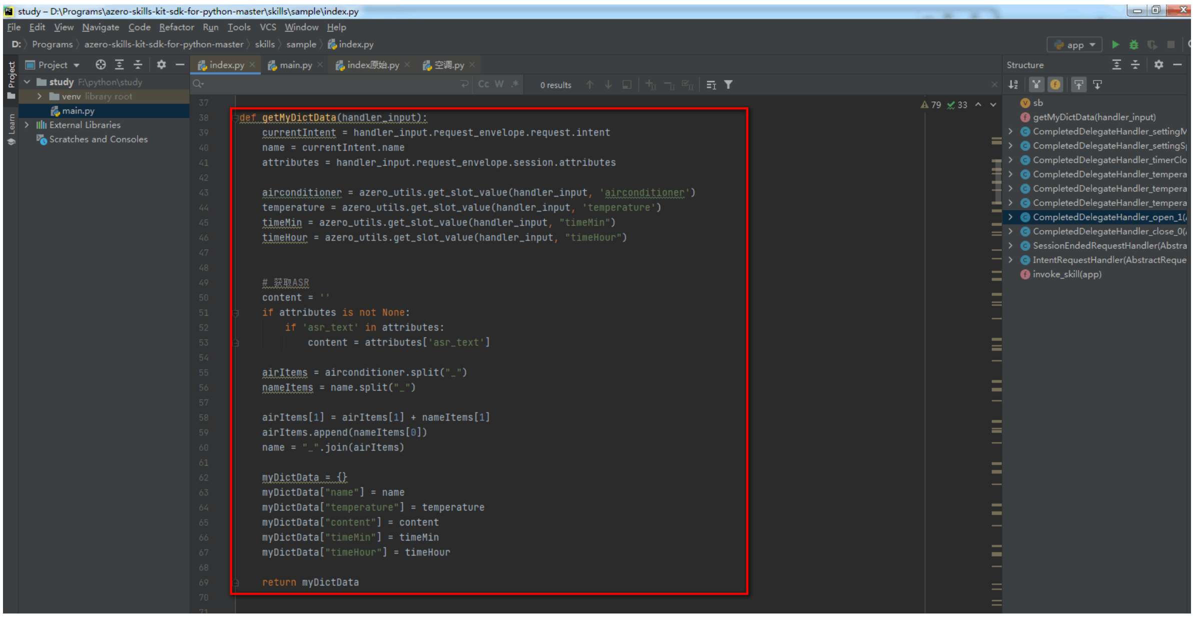
```python
speakOutput = '好哒'
        myDictData = getMyDictData(handler_input)

        return (
            handler_input.response_builder.add_directive(
                RenderTemplateDirective(
                    DefaultTemplateData(
                        ext_content=ExtContent(data=myDictData, type="type", 
tts_text="TTSText", asr_text="ASRText")
                    )
                ))
                .speak(speakOutput)
                .set_should_end_session(True)
                .response
        )
```



```python
class CompletedDelegateHandler_open_1(AbstractRequestHandler):
    def can_handle(self, handler_input):
        return (azero_utils.is_request_type("IntentRequest")(handler_input) and
                azero_utils.is_intent_name("open_1")(handler_input) and
                azero_utils.get_dialog_state(handler_input).value == 'COMPLETED')

    def handle(self, handler_input):
        speakOutput = '好哒'
        myDictData = getMyDictData(handler_input)

        return (
            handler_input.response_builder.add_directive(
                RenderTemplateDirective(
                    DefaultTemplateData(
                        ext_content=ExtContent(data=myDictData, type="type", tts_text="TTSText", asr_text="ASRText")
                    )
                ))
                .speak(speakOutput)
                .set_should_end_session(True)
                .response
        )
```

3- 抽取的getMyDictData方法

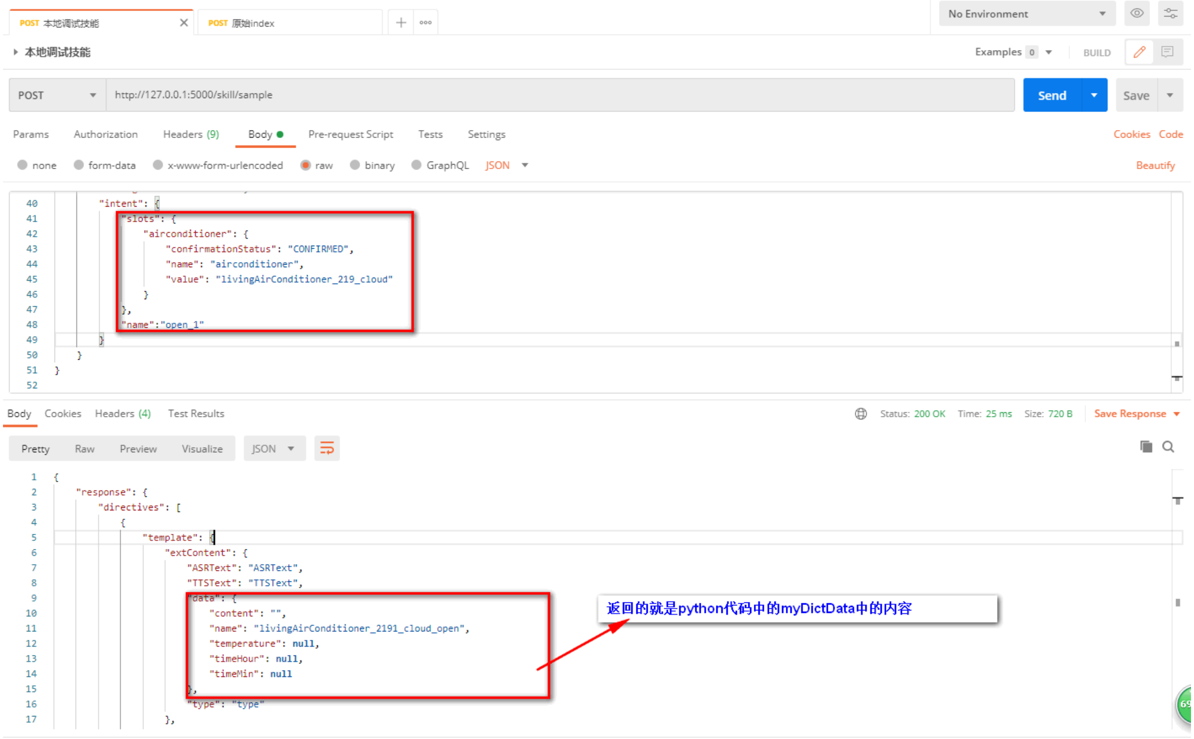
```python
def getMyDictData(handler_input):
    currentIntent = handler_input.request_envelope.request.intent
    name = currentIntent.name
    attributes = handler_input.request_envelope.session.attributes

    airconditioner = azero_utils.get_slot_value(handler_input, 'airconditioner')
    temperature = azero_utils.get_slot_value(handler_input, 'temperature')
    timeMin = azero_utils.get_slot_value(handler_input, "timeMin")
    timeHour = azero_utils.get_slot_value(handler_input, "timeHour")


    # 获取ASR
    content = ''
    if attributes is not None:
        if 'asr_text' in attributes:
            content = attributes['asr_text']

    airItems = airconditioner.split("_")
    nameItems = name.split("_")

    airItems[1] = airItems[1] + nameItems[1]
    airItems.append(nameItems[0])
    name = "_".join(airItems)

    myDictData = {}
    myDictData["name"] = name
    myDictData["temperature"] = temperature
    myDictData["content"] = content
    myDictData["timeMin"] = timeMin
    myDictData["timeHour"] = timeHour

    return myDictData
```


study – D:\Programs\azero-skills-kit-sdk-for-python-master\skills\sample\index.py
File Edit View Navigate Code Refactor Run Tools VCS Window Help
D: 〉 Programs 〉 azero-skills-kit-sdk-for-python-master 〉 skills 〉 sample 〉 index.py
app
Project
index.py
main.py
index原始.py
空调.py
0 results
study F:\python\study
venv library root
main.py
External Libraries
Scratches and Consoles
Structure
sb
getMyDictData(handler_input)
CompletedDelegateHandler_settingM
CompletedDelegateHandler_settingS
CompletedDelegateHandler_timerClo
CompletedDelegateHandler_tempera
CompletedDelegateHandler_tempera
CompletedDelegateHandler_tempera
CompletedDelegateHandler_open_1(
CompletedDelegateHandler_close_0(
SessionEndedRequestHandler(Abstra
IntentRequestHandler(AbstractReque
invoke_skill(app)
79 33

```python
def getMyDictData(handler_input):
    currentIntent = handler_input.request_envelope.request.intent
    name = currentIntent.name
    attributes = handler_input.request_envelope.session.attributes

    airconditioner = azero_utils.get_slot_value(handler_input, 'airconditioner')
    temperature = azero_utils.get_slot_value(handler_input, 'temperature')
    timeMin = azero_utils.get_slot_value(handler_input, "timeMin")
    timeHour = azero_utils.get_slot_value(handler_input, "timeHour")


    # 获取ASR
    content = ''
    if attributes is not None:
        if 'asr_text' in attributes:
            content = attributes['asr_text']

    airItems = airconditioner.split("_")
    nameItems = name.split("_")

    airItems[1] = airItems[1] + nameItems[1]
    airItems.append(nameItems[0])
    name = "_".join(airItems)

    myDictData = {}
    myDictData["name"] = name
    myDictData["temperature"] = temperature
    myDictData["content"] = content
    myDictData["timeMin"] = timeMin
    myDictData["timeHour"] = timeHour

    return myDictData
```

postman调试

```
{
    "version": "1.0",
    "session": {
        "new": true,
        "sessionId": "token.domain-api.session.5e3c1a35d8dafe00060beec8",
        "application": {
            "applicationId": "5e37d715a521820008e9a3f0"
        },
        "user": {
            "userId": "anonymous_2b861255e642461f9d89ad332da6e370"
        },
        "attributes": {
            "source_skill_mapping": {},
            "smarthome_skill_mapping": {},
            "ip": "114.220.24.57"
        }
    },
    "context": {
        "System": {
            "application": {
                "applicationId": "5e37d715a521820008e9a3f0"
            },
            "user": {
                "userId": "anonymous_2b861255e642461f9d89ad332da6e370"
            },
            "device": {
                "deviceId": "52e877f4956ba650dadb3185a3b54746",
                "supportedInterfaces": {
                    "AudioPlayer": {},
                    "Display": {}
                }
            }
        }
    },
    "request": {
        "type": "IntentRequest",
        "requestId": "e4308cca-6e8f-488b-a365-da70453972cf",
        "timestamp": "2020-02-14T00:52:53.541Z",
        "dialogState": "COMPLETED",
        "intent": {
            "slots": {
                "airconditioner": {
                    "confirmationStatus": "CONFIRMED",
                    "name": "airconditioner",
                    "value": "livingAirConditioner_219_cloud"
                }
            },
            "name":"open_1"
        }
    }
}
```



POST http://127.0.0.1:5000/skill/sample

```
        "intent": {
            "slots": {
                "airconditioner": {
                    "confirmationStatus": "CONFIRMED",
                    "name": "airconditioner",
                    "value": "livingAirConditioner_219_cloud"
                }
            },
            "name":"open_1"
        }
    }
}
```

Status: 200 OK Time: 25 ms Size: 720 B

```
{
    "response": {
        "directives": [
            {
                "template": {
                    "extContent": {
                        "ASRText": "ASRText",
                        "TTSText": "TTSText",
                        "data": {
                            "content": "",
                            "name": "livingAirConditioner_2191_cloud_open",
                            "temperature": null,
                            "timeHour": null,
                            "timeMin": null
                        },
                        "type": "type"
                    },
```

返回的就是python代码中的myDictData中的内容