

ChatKitDataAgent

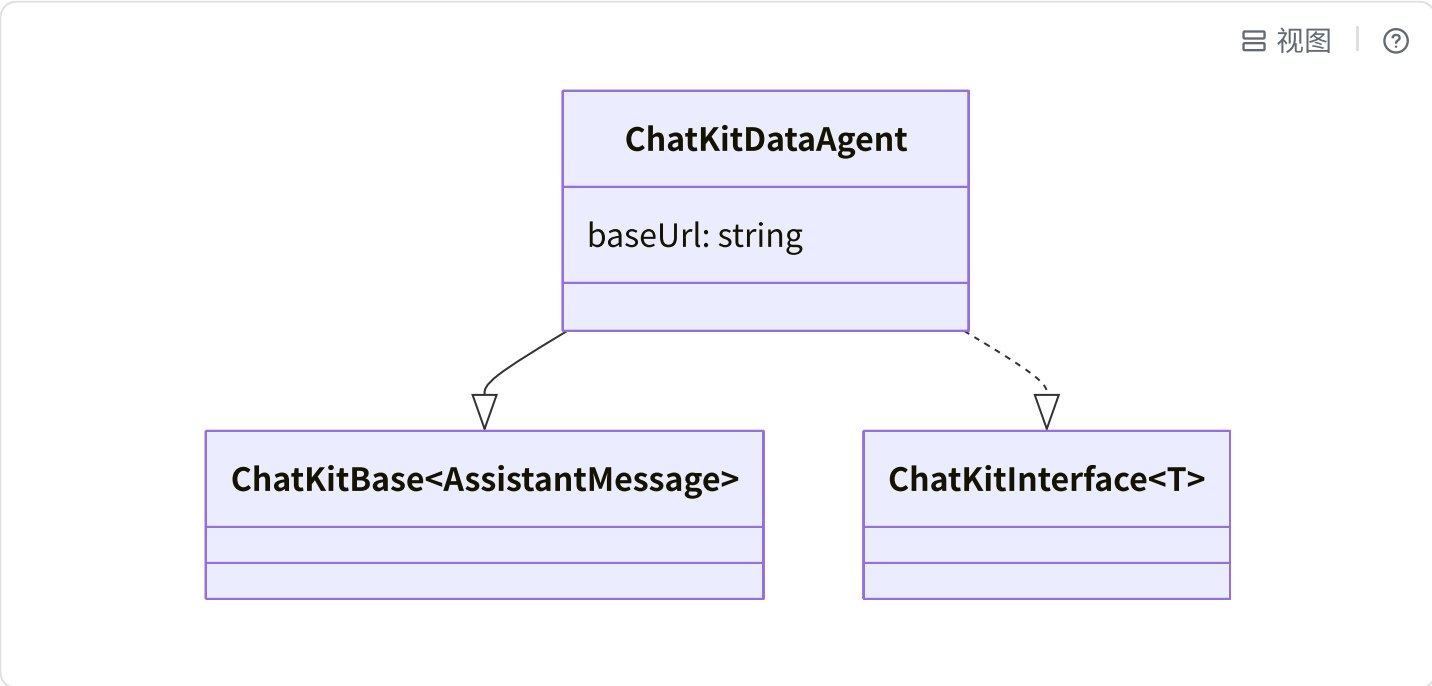
一、基本概念

ChatKitDataAgent 组件是专门适配 AISHU ADP 平台 Data Agent 智能体 API 的对话组件。

二、组件实现

2.1 class ChatKitDataAgent

ChatKitDataAgent 实现 ChatKitInterface 中的接口定义，并继承 ChatKitBase 类。



属性

属性名	类型	说明
baseUrl	string	AISHU Data Agent 平台的访问域名，默认为：/

ChatKitDataAgent 组件实现 ChatKitInterface 以下方法：

- generateConversation()：创建新的会话。
- getOnboardingInfo()：获取会话开场白信息。
- sendMessage()：发送消息给 AI 助手。

- `reduceAssistantMessage()`：从 EventStream 中提取出对 `action` 和 `content`，并根据 `action` 将 `content` 增量更新到 AssistantMessage。
- `shouldRefreshToken()`：判断 API 响应的状态码是否是 401，如果是，则表示需要刷新 Token。

三、处理 EventStream

1、EventStream 的数据结构

EventStream 由多条 Event Message 组成，每条 Event Message 包含一个 `seq_id` 属性用于标记 Event Message 的顺序。

每一条 Event Message 都是一个 JSON 对象，表示一次对 AssistantMessage 对象的更新操作。一条 Event Message 包含 `seq_id`、`key`、`action`、`content` 四个属性：

- `seq_id`：Event Message 序号。
- `key`：要操作的 AssistantMessage 属性的路径的数组表示，需要转换为 JSONPath 后对 AssistantMessage 进行操作。例如：`["message", "content", "middle_answer", "progress", 0]` 转换为 JSONPath 后是 `"message.content.middle_answer.progress[0]"`。
- `action`：表示对 AssistantMessage 执行的操作动作：
 - `upsert` 表示在 JSONPath 路径插入数据
 - `append` 表示在 JSONPath 路径原有数据后追加内容，有两种情况会 `append`：
 - 如果 JSONPath 路径是一个数组下标，则在数组下标位置插入新的对象
 - 否则 JSONPath 路径表示 AssistantMessage 的某个文本类型的属性，在文本后追加内容
 - `end` 表示 EventStream 结束
- `content`：表示要 `upsert` 或 `append` 的内容

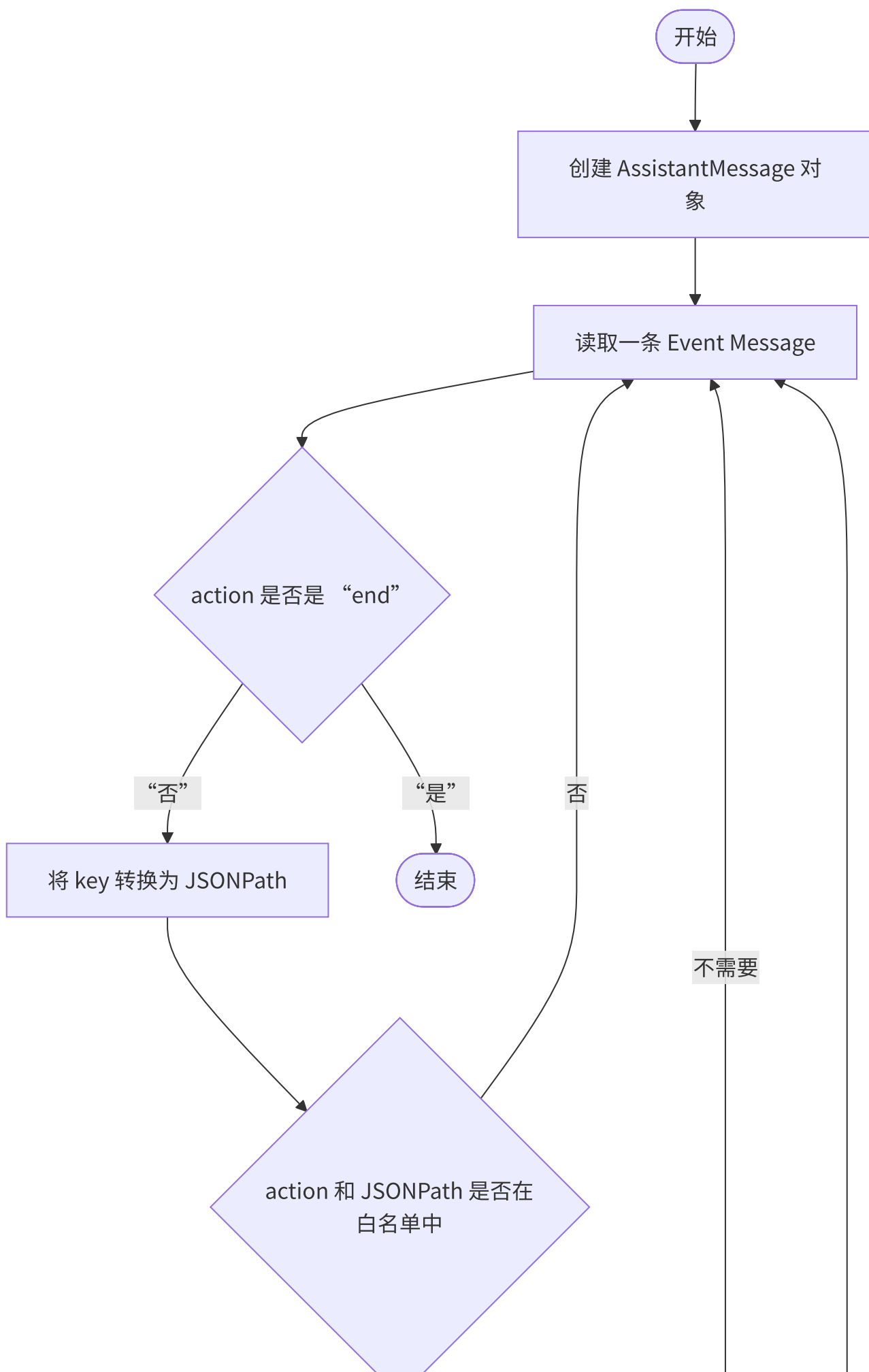
2、AssistantMessage 对象的数据结构

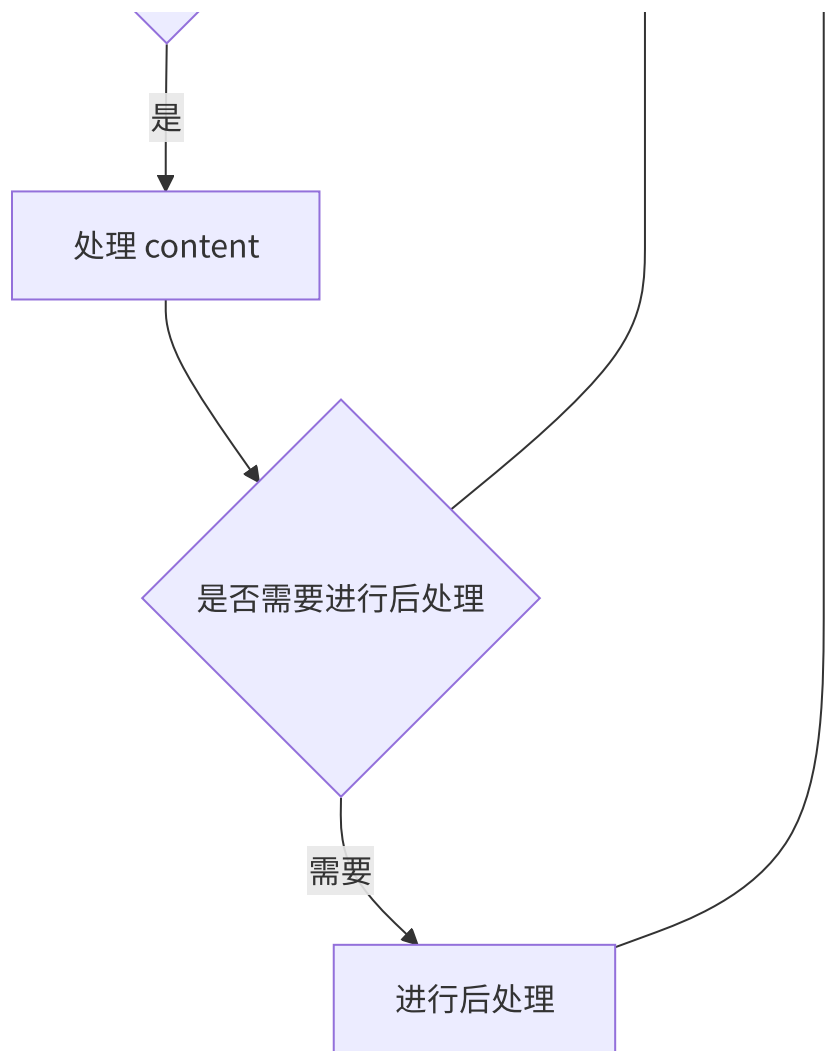
AssistantMessage 对象的数据结构与 `agent-app.schemas.yaml#/components/schemas/Message` 的定义保持一致。

3、处理流程

3.1 流程图

调用 Data Agent 的 `chat/completion` 对话接口时，接口以 EventStream 流的形式输出数据。前端需要不断接收 Event Message 并根据 `action` 来将 `content` 增量更新到 AssistantMessage 中，具体处理流程如下：





3.2 Event Message 白名单

判断 Event Message 包含的 action 和 JSONPath 组合是否在以下名单中：

- 如果在，则根据 action 来更新 AssistantMessage 对象，并在需要时执行后处理流程。
- 如果不在，则跳过该条 Event Message。

action	JSONPath	如何处理 content	后处理
insert	error	将 content 赋值到 JSONPath	不需要
insert	message	将 content 赋值到 JSONPath	不需要
append	message.content.final_answer.answer.text	将 content 追加到 JSONPath 现有内容后	调用 <code>appendTextBlock(message.content.final_answer.answer.text)</code> 将内容输出到界面

<code>app</code> <code>end</code>	<code>message.content.middle</code> <code>e_answer.progress[i]</code>	将 content 赋值到 JSONPath 表示的数组 下标位置	<ul style="list-style-type: none">如果 <code>content.stage</code> 是 “<code>skill</code>”：<ul style="list-style-type: none">如果 <code>content.skill_info.name</code> 是 <code>zhipu_search_tool</code>，调用 <code>appendWebSearchBlock()</code> 将 Web 搜索结果输出到界面否则将 <code>content.skill_info.name</code> 输 出到界面如果 <code>content.stage</code> 是 “<code>llm</code>”，调用 <code>appendTextBlock(message.conte</code> <code>nt.middle_answer.progress[i].a</code> <code>nswer)</code> 将内容输出到界面
<code>app</code> <code>end</code>	<code>message.content.middle</code> <code>e_answer.progress[i].a</code> <code>nswer</code>	将 content 追加到 JSONPath 现有内容后	<ul style="list-style-type: none">调用 <code>appendTextBlock(message.conte</code> <code>nt.middle_answer.progress[i].a</code> <code>nswer)</code> 将内容输出到界面

4、示例

4.1 插入对象

操作前：

代码块

```
1  {}
```

Event Message:

代码块

```
1  {
2    "seq": 0,
3    "key": ["message"],
4    "action": "upsert",
5    "content": {
6      "content": {
7        "final_answer": {
8          "answer": {
```

```

9         "text": ""
10     }
11 },
12     "middle_answer": {
13         "progress": []
14     }
15 }
16 }
17 }

```

操作后：

代码块

```

1  {
2      "message": {
3          "content": {
4              "final_answer": {
5                  "answer": {
6                      "text": ""
7                  }
8              },
9              "middle_answer": {
10                 "progress": []
11             }
12         }
13     }
14 }

```

4.2 追加对象

操作前：

代码块

```

1  {
2      "message": {
3          "content": {
4              "final_answer": {
5                  "answer": {
6                      "text": ""
7                  }
8              },
9              "middle_answer": {
10                 "progress": []
11             }

```

```
12     }
13   }
14 }
```

Event Message:

代码块

```
1  {
2    "seq": 1,
3    "key": ["message", "content", "final_answer", "answer", "text"],
4    "action": "append",
5    "content": "大模型"
6  }
```

操作后:

代码块

```
1  {
2    "message": {
3      "content": {
4        "final_answer": {
5          "answer": {
6            "text": "大模型"
7          }
8        },
9        "middle_answer": {
10         "progress": []
11       }
12     }
13   }
14 }
```

4.3 追加到数组

操作前:

代码块

```
1  {
2    "message": {
3      "content": {
4        "final_answer": {
5          "answer": {
```



```
6         "text": "大模型"
7     }
8 },
9     "middle_answer": {
10         "progress": []
11     }
12 }
13 }
14 }
```

Event Message:

代码块

```
1  {
2      "seq": 2,
3      "key": ["message", "content", "middle_answer", "progress", 0],
4      "action": "append",
5      "content": {
6          "stage": "llm",
7          "answer": "我来帮您"
8      }
9  }
```

操作后:

代码块

```
1  {
2      "message": {
3          "content": {
4              "final_answer": {
5                  "answer": {
6                      "text": "大模型"
7                  }
8              },
9              "middle_answer": {
10                 "progress": [
11                     {
12                         "stage": "llm",
13                         "answer": "我来帮您"
14                     }
15                 ]
16             }
17         }
18     }
```

