autogen_agentchat.messages

AgentMessage

All message types.

alias of `Annotated`[`TextMessage` | `MultiModalMessage` | `StopMessage` | `HandoffMessage` | `ToolCallMessage` | `ToolCallResultMessage`, FieldInfo(annotation=NoneType, required=True, discriminator='type')]

pydantic model BaseMessage

[source]

Bases: `BaseModel`

A base message.

▶ Show JSON schema

Fields:

- `models_usage (autogen_core.models._types.RequestUsage | None)`
- `source (str)`

field **models_usage**: *RequestUsage | None = None*

The model client usage incurred when producing this message.

field **source**: *str [Required]*

The name of the agent that sent this message.

ChatMessage

Messages for agent-to-agent communication.

alias of `Annotated`[`TextMessage` | `MultiModalMessage` | `StopMessage` | `HandoffMessage`, FieldInfo(annotation=NoneType, required=True, discriminator='type')]

pydantic model HandoffMessage

[source]

Bases: `BaseMessage`

A message requesting handoff of a conversation to another agent.

▶ Show JSON schema

Fields:

- `content (str)`
- `target (str)`
- `type (Literal['HandoffMessage'])`

field `content`: *str [Required]*

The handoff message to the target agent.

field `target`: *str [Required]*

The name of the target agent to handoff to.

field `type`: *Literal['HandoffMessage'] = 'HandoffMessage'*

pydantic model `MultiModalMessage`

[source]

Bases: `BaseMessage`

A multimodal message.

▶ Show JSON schema

Fields:

- `content (List[str | autogen_core._image.Image])`
- `type (Literal['MultiModalMessage'])`

field `content`: *List[str | Image] [Required]*

The content of the message.

field `type`: *Literal['MultiModalMessage'] = 'MultiModalMessage'*

pydantic model `StopMessage`

[source]

Bases: `BaseMessage`

A message requesting stop of a conversation.

▶ Show JSON schema

Fields:

- `content (str)`
- `type (Literal['StopMessage'])`

field **content**: *str [Required]*

The content for the stop message.

field **type**: *Literal['StopMessage'] = 'StopMessage'*

pydantic model TextMessage [source]

Bases: `BaseMessage`

A text message.

▶ Show JSON schema

Fields:

- `content (str)`
- `type (Literal['TextMessage'])`

field **content**: *str [Required]*

The content of the message.

field **type**: *Literal['TextMessage'] = 'TextMessage'*

pydantic model ToolCallMessage [source]

Bases: `BaseMessage`

A message signaling the use of tools.

▶ Show JSON schema

Fields:

- `content (List[autogen_core._types.FunctionCall])`
- `type (Literal['ToolCallMessage'])`

field **content**: *List[FunctionCall] [Required]*

The tool calls.

field **type**: *Literal['ToolCallMessage'] = 'ToolCallMessage'*

pydantic model ToolCallResultMessage [source]

Bases: `BaseMessage`

A message signaling the results of tool calls.

▶ Show JSON schema

Fields:

- `content (List[autogen_core.models._types.FunctionExecutionResult])`
- `type (Literal['ToolCallResultMessage'])`

field **content**: *List[FunctionExecutionResult] [Required]*

The tool call results.

field **type**: *Literal['ToolCallResultMessage'] = 'ToolCallResultMessage'*