

腾讯云数据库 AI 服务 Memory 应用指南



腾讯云

【 版权声明 】

©2013–2025 腾讯云版权所有

本文档（含所有文字、数据、图片等内容）完整的著作权归腾讯云计算（北京）有限责任公司单独所有，未经腾讯云事先明确书面许可，任何主体不得以任何形式复制、修改、使用、抄袭、传播本文档全部或部分内容。前述行为构成对腾讯云著作权的侵犯，腾讯云将依法采取措施追究法律责任。

【 商标声明 】



及其它腾讯云服务相关的商标均为腾讯云计算（北京）有限责任公司及其关联公司所有。本文档涉及的第三方主体的商标，依法由权利人所有。未经腾讯云及有关权利人书面许可，任何主体不得以任何方式对前述商标进行使用、复制、修改、传播、抄录等行为，否则将构成对腾讯云及有关权利人商标权的侵犯，腾讯云将依法采取措施追究法律责任。

【 服务声明 】

本文档意在向您介绍腾讯云全部或部分产品、服务的当时的相关概况，部分产品、服务的内容可能不时有所调整。您所购买的腾讯云产品、服务的种类、服务标准等应由您与腾讯云之间的商业合同约定，除非双方另有约定，否则，腾讯云对本文档内容不做任何明示或默示的承诺或保证。

【 联系我们 】

我们致力于为您提供个性化的售前购买咨询服务，及相应的技术售后服务，任何问题请联系 4009100100或 95716。

文档目录

Memory 应用指南

Memory 介绍

关键概念

流程解读

快速体验

控制台指南

新建 Memory

查看 Memory

开发者工具

Python SDK

创建 Client

Session 管理

创建 Session

查询 Session

更新 Session

删除 Session

搜索 Session

Event 管理

追加 Event

查询 Event

删除 Event

State 管理

设置 State

获取 State

获取所有 State

删除 State

清空 State

Record 管理

创建 Record

查询 Record

更新 Record

删除 Record

搜索 Record

HTTP API

API 简介

Session 管理

创建 Session

查询 Session

更新 Session

删除 Session

搜索 Session

Event 管理

追加 Event

查询 Event

删除 Event

State 管理

设置 State

获取 State

获取所有 State

删除 State

清空 State

Record 管理

创建 Record

查询 Record

更新 Record

删除 Record

搜索 Record

Memory 应用指南

Memory 介绍

最近更新时间：2025-09-17 11:02:21

功能介绍

智能体记忆（Memory）旨在为智能体（Agent）提供长期记忆与上下文管理能力。它能够在多轮对话和跨会话的场景中，自动捕捉用户关键信息、任务进度和经验总结，并将这些信息存储为可检索的长期记忆。通过使用 Memory，智能体不仅能“回答问题”，还能“记住用户”，并在后续交互中主动回忆与应用，从而实现更个性化、更连续、更智慧的交互体验。

核心价值

- 上下文延续：在跨会话场景下保持用户历史信息，避免用户重复输入。
- 个性化服务：基于长期存储的用户画像与偏好，为用户提供定制化推荐与建议。
- 任务追踪：记录用户的待办事项、进度与计划，支持跨会话的任务管理与提醒。
- 知识沉淀：将用户对话中的事实与经验转化为可复用的知识，辅助智能体更高效地推理与决策。

设计框架

Memory 模块是 Agent 实现连续、个性化对话的智能核心。当用户（Actor）与 Agent 产生交互时，原始的对话内容将作为事件（Event）被记录下来，形成短期记忆。随后，记忆策略（Strategy）会根据配置，在特定时机（如会话暂停或达到一定对话长度）自动触发提取逻辑，调用 LLM 对原始的、冗长的短期记忆进行深度处理，提炼出人物画像，任务信息，事实经验等高维度的信息。这些信息被转化为长期记忆（Record），并持久化存储。



基本概念	含义	资源限制说明
Memory	智能体记忆，一个专门用来存储历史对话、用户偏好、事件记录等信息的“记忆库”。	每个腾讯云账户下最多可创建5个Memory，用于隔离不同应用或核心场景。
Actor	用户，指 Agent 对话中的一个终端用户。	每个 Memory 内最多可定义500个不同的用户、设备或智能体角色。
Session	会话，用户就一个话题或多个话题而新建的会话窗口。	每个 Actor 在其生命周期内最多可创建1000个独立会话。
Event	短期记忆，一轮对话或者多轮对话的具体内容、工具使用或其他相关信息。	每个 Memory 下总短期记忆条数上限为100万条。
State	状态记忆，一个基于键值对 (KV) 的临时存储系统，专门用于维护一个 Session (会话) 在生命周期内的各种临时状态、上下文和中间数据。	每个会话中最多可保存100个状态键值对，用于记录上下文的变化。

<p>Record</p>	<p>长期记忆，基于 Event 会话数据，通过 LLM 提取的事实、画像、摘要等可供长期复用的结构化知识核心。</p>	<p>每个 Memory 下总长期记忆条数上限为 1000 万条。</p>
<p>Strategy</p>	<p>长期记忆策略，从用户对话 Event 中提取生成长期记忆的策略。</p>	<p>无限制，支持的策略如下：</p> <ul style="list-style-type: none"> ● 人物画像：通过大模型提取原文对话中的人物身份、社交关系、行为特征与个性偏好等信息，并整合成语义化的用户画像。 ● 任务信息：通过大模型提取会话级的任务概览与进度信息，并进行会话级的持续更新，实现任务上下文的有效压缩，为长任务提供有效的任务信息。 ● 事实经验：通过大模型提取原文对话中的经历事件、知识见解、操作流程与经验总结等信息，并整合成语义化的事实记忆。 ● 自定义策略：允许用户根据特定领域、场景或需求，定制专属记忆生成策略。

使用场景

Memory 模块是智能体（Agent）实现持续认知与个性化交互的核心基础设施。Agent 开发者可以根据不同的上下文需求，从短期记忆（Event）中获取最近的对话原文，从长期记忆（Record）中精准检索出被提炼的信息，从状态记忆（State）中获取对话窗口中临时的信息。通过使用不同的记忆组合，LLM 不仅能理解当前的直接问题，还能拥有更完整的背景信息和用户历史等信息，从而生成更加连贯、精准且个性化的回应，完成从记忆到智能的完美闭环。如下是一个旅游咨询的一个使用场景。

Session

我下周要去杭州三天，你能帮我安排一个行程吗？ Event 1

好的。在安排行程前，我想确认一下，您是喜欢热闹的点，还是安静一些的地方呢？

我更喜欢人少安静的环境。像上次去的云栖竹径，上午人很少，体验特别好。

原来如此，这类经验我会帮您记录下来。您的饮食偏好呢？

我喜欢本帮菜，也会特地去看老字号餐馆。比如奎元馆的虾爆鳝面，我以前赶早去过，排队没那么久。

Event 2 明白，那我在行程里会推荐一些地道餐馆。顺便确认一下，您对住宿有什么要求？

最好是安静的酒店，而且带健身房会更好。

好的。另外，您是独自出行吗？

我会和女朋友一起去，她特别喜欢现代艺术展览。

好的。那我帮您先拟一个基础行程。第一天：杨公堤、茅家埠 → 奎元馆午餐 → 三潭印月 → 天目里美术馆 → 傍晚湖边茶社 → 晚餐胜利河美食街。第二天：灵隐寺 → 法喜寺 → 新周记午餐 → 浙江省博物馆之江馆 → 晚餐新丰小吃。第三天：西溪湿地一日游。

Event 3 听起来不错。第三天适当加一个在咖啡馆的休息时间，平衡下节奏，不会太累。

好主意！那我在第三天下午的行程里加入小河直街的咖啡馆，让整体节奏更轻松。

3 小时后

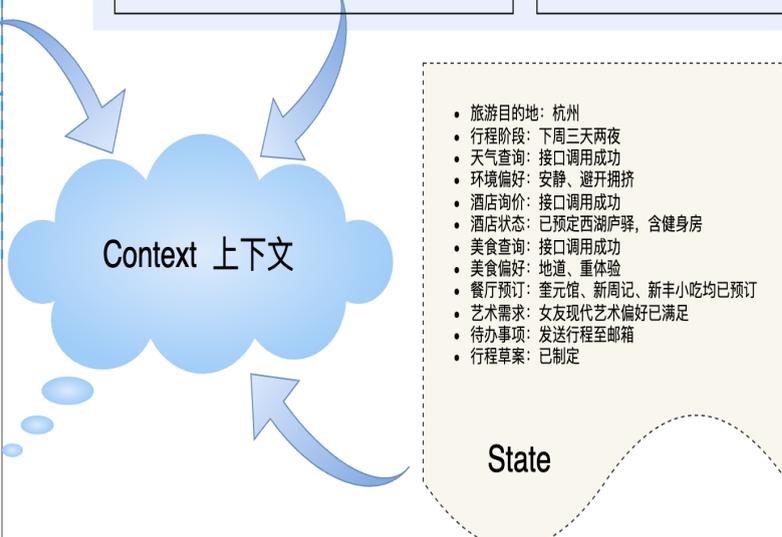
我上次说的去杭州的行程，酒店预定好了吗？

🔍 在思考中.....

欢迎回来！您的行程都已安排妥当。酒店已预订‘西湖庐驿’。餐厅方面：‘奎元馆’（蟹粉面）和‘新周记’（创意杭帮菜）均已预订。需要我把完整行程表发送到您的邮箱吗？

Record

<p style="text-align: center;">事实经验</p> <ol style="list-style-type: none"> 1. 用户上次去云栖竹径，发现上午人少更安静。 2. 用户曾在奎元馆吃虾爆鳝面，早点去能减少排队。 3. 用户总结安静的景点体验更放松。 4. 用户认为旅行中加入咖啡馆休息能平衡节奏。 	<p style="text-align: center;">人物画像</p> <ol style="list-style-type: none"> 1. 用户喜欢人少安静的环境，倾向避开热门景点。 2. 用户对美食有强烈兴趣，尤其偏好本帮菜与老字号餐馆。 3. 用户在酒店选择上注重健身设施和安静氛围。 4. 用户的伴侣对现代艺术展览有浓厚兴趣。
<p style="text-align: center;">任务信息</p> <ol style="list-style-type: none"> 1. 用户计划下周在杭州进行为期三天的旅行。 2. 第一天：杨公堤、茅家埠 → 奎元馆午餐 → 三潭印月 → 天目里美术馆 → 傍晚湖边茶社 → 晚餐胜利河美食街。 3. 第二天行程：灵隐寺 → 法喜寺 → 新周记午餐 → 浙江省博物馆之江馆 → 晚餐新丰小吃。 4. 第三天行程：西溪湿地摇橹船 → 小河直街咖啡馆休憩。 	<p style="text-align: center;">自定义 请总结用户对旅游的关注维度</p> <ol style="list-style-type: none"> 1. 环境维度为安静人少、避开拥挤，追求慢节奏与自然融合。 2. 美食维度为地道风味优先，尽量减少排队时间。 3. 设施维度住宿必须配备健身房，保障日常运动需求。 4. 伴侣维度，满足同行女友对现代艺术的兴趣，希望纳入行程。



情感陪伴场景

在情感陪伴场景中，Memory 模块扮演着智能体的“情感中枢”，可超越简单地记录对话内容，转而深度理解并记忆用户的情绪波动、性格特质、生活事件及其背后的情感脉络。系统会自动化地从日常交流中提炼出用户的压力源、安慰方式、情绪周期模式以及重要的人际关系，并将其沉淀为高维度的情感画像。当用户再次交互时，智能体能够瞬时调用这些记忆，提供具有高度共情性和连续性的回应。例如，它能主动识别出用户周日晚上的焦虑情绪，并回忆起此前有效的舒缓方式，从而提供“感同身受”的安慰与建议。

医疗与健康管理

Memory 可以作为一个持续、私密的个人健康助手，通过长期、细致地记录用户的健康信息，提供高度个性化的支持。Memory 能够帮助管理慢性病，记忆用户的用药历史、症状变化及对特定药物的反应，从而提供更精准的服药提醒和生活方式建议（例如：“您上次提到吃了 X 食物后血糖会升高，这次需要记录一下吗？”）。同时，Memory 在心理健康方面，通过追踪长期的情绪波动模式，识别可能的诱因（如季节变化或特定事件），并在用户

情绪低落时给予贴合其个人历史的应对策略。此外，作为健康档案助手，Memory 还能帮助用户轻松记录和整理零散的健康信息，如历年体检指标变化、过敏史和家族病史，并在需要时生成简洁清晰的摘要，极大方便与医生的高效沟通。

个性化教育场景

在个性化教育领域，Memory 模块是驱动智能导师实现“因材施教”的“智能学籍卡”，可持续追踪并分析每一位学习者的完整互动轨迹，精准刻画其知识掌握度、思维误区、学习节奏与内容偏好，动态构建出一个不断演进的个人能力模型。基于此模型，教学智能体能够进行前瞻性的干预与自适应调整：在讲解新知识点时，自动关联其已掌握的概念进行正向迁移；在练习环节，精准预判并提示其可能出现的特定错误；在内容呈现上，优先采用该学习者最容易理解的案例或形式（如图解、类比等）。使得教学互动不再是千篇一律的重复，而是完全围绕学习者独特性展开的动态、高效、正向反馈的成长旅程，最大化提升学习者的信心与学习效果。

关键概念

最近更新时间：2025-09-17 14:36:52

本文档旨在阐述智能对话系统中的五大核心概念：Session（会话）、Event（事件）、State（状态）、Record（记忆）和 Strategy（策略）。从定义、核心特性与场景示例三个维度对每个概念进行详细说明，以帮助您深入理解其设计理念、运作机制与应用方式。

会话 Session

维度	详细说明
定义	Session 是管理持续对话与维持上下文的独立会话单元，就像一个独立的聊天窗口，其将所有关联的交互内容归集于一处，可无缝续接与历史追溯。
核心特性	<ul style="list-style-type: none">创建于每一次新的对话开始时，每个新会话窗口都会创建一个独立且唯一的 Session，实现不同对话上下文之间的隔离。自动归集并永久保存窗口内的所有对话，确保上下文不丢失，支持无缝的“原地续聊”。提供对会话的创建、查询、更新和检索等完整管理功能，并记录包括起止时间、所属应用等元数据，便于审计与追溯。支持通过 Session 名称等规则进行快速搜索与筛选，帮助用户从大量历史会话中精准定位目标内容。
场景示例	<p>用户小明计划一次杭州旅行。他新建了一个 Session，并命名为“杭州三日游计划”。</p> <ul style="list-style-type: none">他输入第一句话：“帮我规划一下杭州三天的行程。”智能体回复：“好的，请问您的预算和偏好是什么？”小明继续提问：“预算5000元，喜欢人文历史。”（对话继续...）第二天，小明搜索 Session 名称“杭州”，查找到这个 Session 并打开，之前的全部对话（所有 Events）都完整呈现，他可以接着上次的进度继续会话。

事件 Event

维度	详细说明
定义	Event 是会话中记录原子交互操作的独立事件单元，就像聊天记录里的一条完整消息或一个操作日志，其精准捕获对话过程中的每一次请求与响应，支撑全链路追溯与回放。
核心特性	<ul style="list-style-type: none">发生于每一次用户输入、助手回复或工具调用时。具备唯一序列 ID、明确角色（用户/助手/工具）、时间戳及内容负载。支持按发生时间顺序排列，结构化组成会话流。支持按角色、事件类型、关键词、时间范围等多维度检索与分析，用于审计与溯源。

场景示例	<p>用户小明计划一次旅行，如下对话过程，生成一个Event。</p> <ul style="list-style-type: none"> ● 他输入首句话：“帮我规划杭州三天的行程。” <ul style="list-style-type: none"> ○ 智能体回复询问预算与偏好。 ○ 小明回复：“预算5000元，喜欢人文历史。” ● 智能体调用“行程生成工具”进行查询。 <ul style="list-style-type: none"> ○ 工具返回结果，智能体组织回复。
-------------	---

状态 State

维度	详细说明
定义	State 是会话中的“上下文快照”，用于记录对话的当前进度与执行环境，明确下一步操作的上下文依据。
核心特性	<ul style="list-style-type: none"> ● 记录用户意图、任务进度、临时结果、工具会话句柄等关键要素，通常以结构化键值存储。 ● 提供类似“游戏存档”的续话能力，支持中断后无缝恢复，避免重复背景信息。 ● 确保智能体在用户返回会话时仍能准确理解任务进度、参数约束与历史上下文。
场景示例	<p>用户小明在“杭州三日游” Session 中规划行程：</p> <ul style="list-style-type: none"> ● 他设定预算为5000元（更新 State 中的“预算”字段）。 ● 他表明偏好“人文历史”（State 中“偏好”字段被刷新）。 ● 智能体调用行程工具，生成初步方案草稿（State 中记录“草稿行程”及“工具会话句柄”）。 <p>中途小明切换会话处理其他事务，第二天再次返回：</p> <ul style="list-style-type: none"> ● 系统依据上一次的 State 记录自动恢复上下文，智能体清楚记得预算、偏好及草稿进度。 ● 小明无需重复前提条件，可直接基于现有方案继续优化，实现无缝续聊。

长期记忆 Record

维度	详细说明
定义	Record 是按照记忆策略提炼与整合后形成的长期可复用知识单元，用于支持跨会话的个性化知识检索与一致性交互。
核心特性	<ul style="list-style-type: none"> ● 源于 Event 的深度提炼，经过去重、合并与结构化处理，形成可复用的知识单元。 ● 记录包括事实结论、人物画像、任务结果、操作流程、关键约束等高价值信息。 ● 每条 Record 均保留来源追溯信息，包括原始 Session、Event 及策略版本与时间戳。 ● 支持基于语义召回匹配的 Record，且可动态注入 Record，提升交互准确性与效率。
场景示例	<p>用户小明曾在多个会话中规划旅行并表达偏好：</p> <ul style="list-style-type: none"> ● 在“杭州行” Session 中，智能体从其对话中提取“偏好博物馆”并生成一条 Record。

- 在“周末安排” Session 中，小明提到“不爱拥挤景点”，系统再次生成一条约束类 Record。

当小明新建“南京文化之旅”会话时：

- 系统自动从记忆库中语义检索到之前生成的“偏好博物馆”与“避拥挤”等 Records。
- 这些 Records 被注入新会话上下文，智能体在推荐景点时自动排除拥挤景点，并优先建议博物馆路线。
- 小明无需重复说明兴趣与禁忌，即可获得高度个性化的连贯体验。
- 所有推荐均可通过 Record 追溯至原始会话和事件，保障信息透明与可解释性。

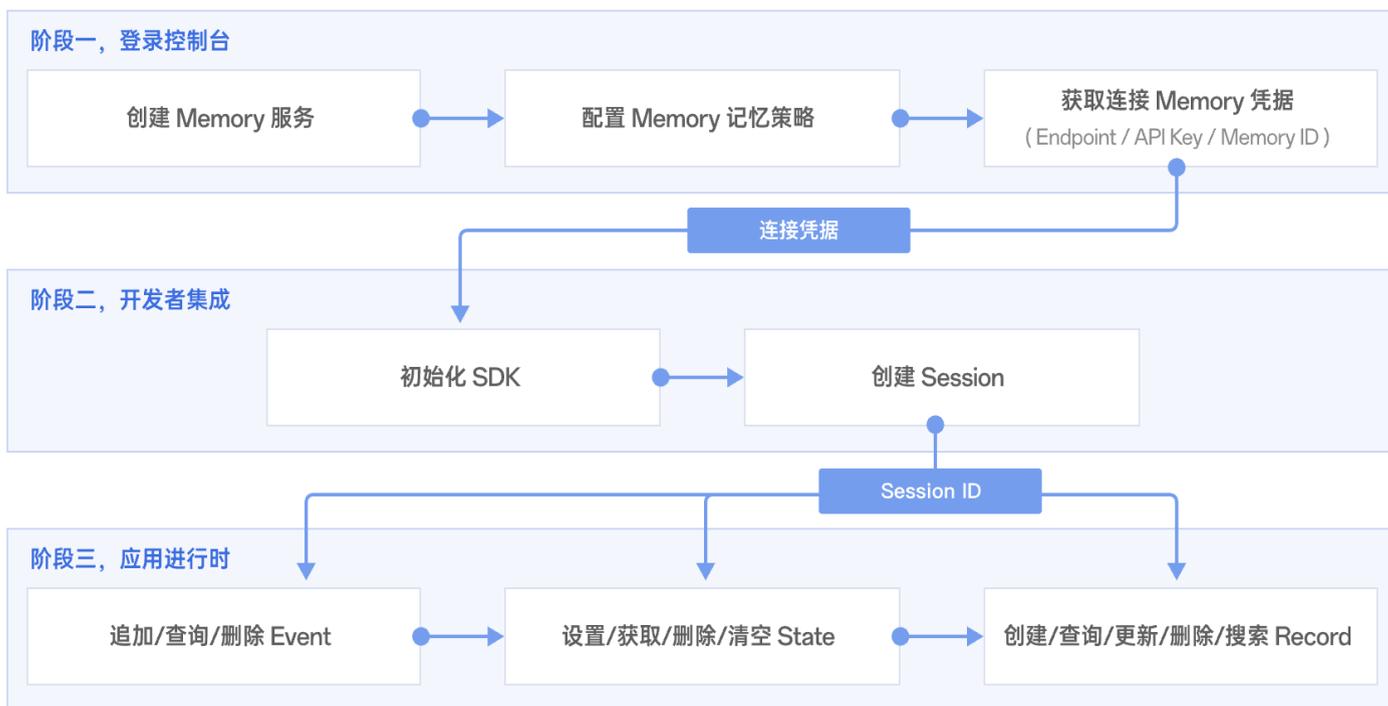
策略 Strategy

维度	详细说明
定义	Strategy 是将短期对话内容 (Event) 转化为长期可复用记忆 (Record) 的提炼与整合规则，是实现记忆智能化管理的核心配置引擎。
核心特性	<ul style="list-style-type: none"> 提供从 Event 中抽取关键信息，并经过去重、合并、结构化处理，最终产出包含结构化字段并可附带语义向量的 Record。 支持多种触发机制：包括工具调用完成、任务阶段变更、定时运行、置信度阈值触发以及人工手动触发。 每次执行均记录策略标识、版本、来源会话与事件等元数据，确保生成过程可追溯、可审计。 提供如下三类基础记忆策略，并支持用户通过配置提示词自定义记忆生成规则，适应多样化场景需求。 <ul style="list-style-type: none"> 人物画像策略：提取并整合对话中的人物身份、社交关系、行为特征与个性偏好，构建语义化的用户画像。 任务信息策略：提取任务概览与进度信息，实现会话级任务上下文的持续压缩与更新，为长任务提供有效信息支撑。 事实经验策略：提取经历事件、知识见解、操作流程与经验总结，形成可复用的语义化事实记忆。 开启合适的策略后，系统会自动总结关键信息（如偏好、任务进度、复盘经验），下次对话可被智能召回，满足用户个性化服务需求。
场景示例	<p>用户小明在多次会话中讨论智能家居配置：</p> <ul style="list-style-type: none"> 当他完成一次设备调试后，任务变更事件触发任务信息策略，系统自动生成一条“家庭网络配置完成”的 Record。 当他多次选择“夜间自动调节亮度柔和”时，人物画像策略被触发，生成“偏好柔和灯光”的画像 Record。 <p>当小明在新会话中说“调整客厅灯光”时：</p> <ul style="list-style-type: none"> 系统自动召回“偏好柔和灯光”和“网络配置完成”等 Records，智能体直接给出个性化建议，无需重复询问背景。 所有 Record 均带有策略版本和来源事件标识，确保记忆的准确性和可追溯性。

流程解读

最近更新时间：2025-09-12 21:41:51

记忆库的构建是一项融合了策略设计、服务配置与开发集成的系统工程。本文将从零到一全面解读整个构建的全流程，提供清晰框架。



序号	阶段	操作过程
1	准备阶段	1. 创建 Memory 实例，并设置记忆策略。具体操作，请参见 新建 Memory 。 2. 获取 Memory 连接凭据。具体操作，请参见 查看 Memory 。 <ul style="list-style-type: none"> ○ Endpoint：Memory 服务访问地址。 ○ API Key：访问认证密钥。 ○ Memory ID：系统随机分配的具有唯一标识的服务 ID。
2	集成阶段	1. 获取 Python SDK 安装包 (tdaimemory-0.0.1-py3-none-any.whl)，执行如下命令即可完成安装。 <pre style="background-color: #333; color: #fff; padding: 10px; border-radius: 5px;">pip3 install tdaimemory-0.0.1-py3-none-any.whl</pre> 2. 初始化 SDK，创建 Memory 的客户端对象。具体接口，请参见 创建 Client 。 3. 创建一个新会话，并将其关联到指定的记忆库。具体接口，请参见 创建 Session 。

		<p>4. (可选) 可根据需要查询/更新/删除/搜索 Session。更多接口, 请参见 Session 管理。</p>
<p>3</p>	<p>应用程序运行时</p>	<ol style="list-style-type: none"> 1. Event 接口, 支持向指定的会话中添加新的对话事件, 并将此事件持久化至记忆库中。同时, 该接口亦提供对应事件的查询与删除功能, 满足数据管理需求。具体接口, 请参见 Event 管理。 2. Record 接口, 支持对记忆记录的直接操作能力: 搜索、查询, 新建、更新与删除。具体接口, 请参见 Record 管理。 3. State 接口, 可设置当前会话中的状态数据, 支持查询、删除、清空状态数据。具体接口, 请参见 State 管理。

快速体验

最近更新时间：2025-09-17 16:17:35

本文旨在指导您如何快速部署并体验 Memory 服务，包括开通服务、连接服务、写入事件及检索记忆等，通过真实案例演示其快速构建过程。

说明：
当前 Memory 为内测阶段，免费体验时长为3个月。

准备工作

- 已注册腾讯云账号并完成实名认证。
 - 如需注册腾讯云账号：请单击 [注册腾讯云账号](#)。
 - 如需完成实名认证：请单击 [实名认证](#)。
- 申请一台与 Memory 服务在同一地域（目前仅开放广州）的云服务器，并确保该服务器与 Memory 网络互通，以获得安全且低延迟的访问体验。

操作步骤

步骤1：新建 Memory

1. 使用腾讯云账号登录 [Memory 智能体控制台](#) 页面，单击**新建 Memory**，进入 Memory 配置页面，如下图所示。

← 新建 Memory
白 免费体验时长: 3个月

基本信息

地域: 广州 v
当前仅开放广州地域, 其他地域敬请期待或联系商务

服务名称: Travel_Consultation ✔
支持 60 个字符内的中英文、数字、中划线及下划线

描述:

用于描述记忆体的用途、场景或背景

0 / 200
支持 200 个字符内的中英文、数字、特殊字符: _-+&!@#%^^*()

关联大模型: 通用大模型 限时 Token 免费体验
使用大模型并搭配默认或自定义的提示词来实现长期记忆的提取

标签: 标签键 标签值 ✕
+ 添加 📄 键值粘贴板
如现有的标签不合适, 您可以去控制台[新建标签/标签值](#)

记忆配置

短期记忆容量: - 100 + 万条

长期记忆策略:

👤 **人物画像**

通过大模型提取原文对话中的人物身份、社交关系、行为特征与个性偏好等信息, 并整合成语义化的用户画像。

个性化推荐 用户画像构建 智能助理个性调优

📋 **任务信息**

通过大模型提取会话级的任务概览与进度信息, 并进行会话级的持续更新, 实现任务上下文的有效压缩, 为长任务提供有...

任务上下文压缩 长任务执行支持 多任务协同调度

📚 **事实经验**

通过大模型提取原文对话中的经历事件、知识见解、操作流程与经验总结等信息, 并整合成语义化的事实记忆。

知识库沉淀 经验复用 培训与决策支持

+ 自定义策略 (还可以添加 3 个)

2. 参见下表说明, 指定**服务名称**以及**长期记忆策略**, 单击**立即试用**, 等待任务执行完成, 提示创建成功即可。

界面参数	解释	限制与建议
服务名称	记忆体的唯一标识名称。	支持60个字符内的中英文、数字、中划线及下划线, 例如: Travel_Consultation。
长期记忆策略	指定长期记忆策略。	勾选默认策略: 任务画像、任务信息、事实经验。

步骤2: 获取 Memory 连接凭据

在 Memory 服务列表页面, 可看到新建的 Memory 卡片, 如下图所示。

- **Memory ID:** 将鼠标悬停在 ID 上, 点击 ID 可一键复制 Memory ID。
- **访问地址及密钥:** 单击卡片, 可在**服务概览**的 **API 接入**区域获取 Memory 访问地址与访问密钥。



步骤3: 在客户端安装 tdaimemory SDK

使用 pip 命令，在 CVM 上安装 SDK，如下示例，推荐使用 Python 3.8及以上版本。

```
pip3 install tdaimemory-0.0.1-py3-none-any.whl
```

步骤4: 初始化 Memory 客户端

配置连接凭据参数，创建客户端连接对象。获取连接凭据，请参见 [步骤2: 获取 Memory 连接凭据](#)。

- **EndPoint:** 配置新建 Memory 的访问地址。
- **ApiKey:** 配置新建 Memory 的访问密钥。
- **MemoryId:** 配置 Memory ID。

```
from tdaimemory import MemoryClient

EndPoint = "https://memory.tdai.tencentyun.com"
ApiKey = "*****"
MemoryId = "tdai-mem-mfdk****"

client = MemoryClient(endpoint=EndPoint,
                      api_key=ApiKey,
                      memory_id=MemoryId,
                      default_actor_id="user-initial-test")
```

步骤5: 创建 Session

调用 `create_session()` 接口来为特定用户 (actor) 创建一个新的 Agent-Memory 会话。创建时需指定一个会话名称 (name)，该名称可为1到128个字符之间的任意字符串。

```
import json
session = client.create_session(
    name="Travel_Consultation",
)
session_id = session['session_id']
print(json.dumps(session_id, ensure_ascii=False))
```

步骤6: 追加 Event

在创建的 Session 后，使用 `append_event()` 接口为此次会话追加具体的事件，系统将按照记忆策略分析写入记忆库。

```
event = client.append_event(  
    session_id=session_id,  
    messages={  
        "chat": [  
            {  
                "role": "user",  
                "content": "嗨，帮我规划一个下周去杭州的3天行程。"  
            },  
            {  
                "role": "assistant",  
                "content": "好的，已为您推荐杭州三日行程。第一天，我们聚焦西湖经典。清晨从断桥残雪出发，漫步白堤，感受水墨画般的湖光山色。随后乘船登岛探访三潭印月，下午登雷峰塔俯瞰全景，傍晚在曲院风荷赏夕阳，晚上则去河坊街夜市品尝地道小吃。\\n\\n第二天，探索湿地与运河文化。上午在西溪湿地乘坐摇橹船，听船娘小调，感受水乡慢生活；下午逛一逛桥西历史街区和文艺的小河直街，参观工艺美术博物馆，在咖啡馆小憩后即可轻松结束旅程，准备返程。\\n\\n第三天，体验禅意与宋韵。上午前往灵隐寺景区祈福，在法喜寺的黄墙拍照打卡；下午沉浸于宋城，体验AR剧本杀并观看《宋城千古情》演出；如果傍晚还有精力，可以去九溪烟树享受清凉的溪水。"  
            },  
            {  
                "role": "user",  
                "content": "我喜欢人少、安静的地方，对美食特别感兴趣。"  
            },  
            {  
                "role": "assistant",  
                "content": "基于您喜欢安静和美食的偏好，我为您优化了行程，避开了团队游客扎堆的景点，专注于幽静的体验和地道的风味。\\n\\n第一天：避世西湖\\n清晨漫步杨公堤与茅家埠，享受静谧湖景。午餐在老字号「奎元馆」吃虾爆鳝面。下午乘船访三潭印月，傍晚在湖边茶社小坐。晚餐去本地人爱的胜利河美食街尝地道风味。\\n\\n第二天：禅意之路\\n一早探灵隐寺，然后沿幽静天竺路步行至法喜寺。午餐在「知竹」吃素面。下午深入云栖竹径看竹海。晚餐用「新丰小吃」的喉口馒头等市井味道收尾。\\n\\n第三天：水乡慢生活\\n全天在西溪湿地坐摇橹船，深入芦苇荡享受宁静。午后往文艺的小河直街，在运河畔的咖啡馆小坐休憩，结束旅程。"  
            },  
            {  
                "role": "user",  
                "content": "太好了，酒店要带健身房，另外我对花生严重过敏，预订餐厅时请务必注意。"  
            },  
            {  
                "role": "assistant",  
                "content": "好的，已完全了解您的需求（偏好安静人少、喜爱美食、酒店需带健身房、对花生严重过敏）。为您精选的酒店（带健身房）：\\n\\n西湖庐驿精品酒店：位于
```

```
西湖区，设计静谧，配备健身房，且远离闹市，符合您对安静的需求。\\n\\n杭州西溪紫金港希尔顿逸林酒店：靠近西溪湿地，设施完善，健身房专业，适合在第三天安排湿地行程时入住。"
```

```
    },  
    {  
      "role": "user",  
      "content": "完美！哦对了，我女朋友更喜欢现代艺术，有什么地方可以推荐吗？"  
    },  
    {  
      "role": "assistant",  
      "content": "收到！考虑到您女朋友的喜好。这是为您和您女友定制的杭州三日行程：\\n\\n第一天：上午游茅家埠、杨公堤；下午观天目里美术馆现代展览。晚餐于胜利河美食街 品尝安全地道的本帮菜。\\n\\n第二天：上午早访灵隐寺，漫步幽静天竺路。下午重点参观浙江省博物馆之江馆 重磅国际艺术大展。晚餐体验本地 新丰小吃（已规避风险）。\\n\\n第三天：上午于 西溪湿地乘摇橹船。下午在小河直街 咖啡馆悠闲小憩，结束旅程。"  
    }  
  ],  
}  
)
```

步骤7：搜索记忆

❗ 说明：

等待 Event 写入完成，即可开始搜索 Record 记忆。

```
search_records = client.search_records(  
    content="下周去杭州的行程会去哪些地方？",  
    session_id=session_id,  
    # strategies=["Factual_Experience"],  
)  
print(json.dumps(search_records, ensure_ascii=False))
```

搜索结果，如下所示。

```
{  
  "records": [  
    {  
      "created_at": 1757907027,  
      "event_ids": "[\"event-y4YD****\"]",  
      "record_content": "第一天：上午游茅家埠、杨公堤；下午观天目里美术馆现代展览。晚餐于胜利河美食街 品尝安全地道的本帮菜。",  
    }  
  ]  
}
```

```
"record_id": "record-jTqG****",
"score": 0.7354369,
"strategy_name": "Task_Information",
"updated_at": 1757907027
},
{
  "created_at": 1757907016,
  "event_ids": "[\"event-y4YD****\"]",
  "record_content": "第二天: 上午早访灵隐寺, 漫步幽静天竺路。下午重点参观 浙江省博物馆之江馆 重磅国际艺术大展。晚餐体验本地新丰小吃",
  "record_id": "record-YqNmt****",
  "score": 0.6929885,
  "strategy_name": "Task_Information",
  "updated_at": 1757907016
},
{
  "created_at": 1757907009,
  "event_ids": "[\"event-y4YD****\"]",
  "record_content": "第三天上午于西溪湿地乘摇橹船。下午在小河直街 咖啡馆悠闲小憩。",
  "record_id": "record-FtDf****",
  "score": 0.64288217,
  "strategy_name": "Task_Information",
  "updated_at": 1757907009
},
{
  "created_at": 1757907020,
  "event_ids": "[\"event-y4YD****\"]",
  "record_content": "前两晚入住西湖庐驿精品酒店, 位于西湖区, 设计静谧, 配备健身房, 且远离闹市",
  "record_id": "record-Zp9Ji0****",
  "score": 0.5674117,
  "strategy_name": "Factual_Experience",
  "updated_at": 1757907020
},
{
  "created_at": 1757907022,
  "event_ids": "[\"event-y4YD****\"]",
  "record_content": "第三天晚上入住杭州西溪紫金港希尔顿逸林酒店, 靠近西溪湿地, 设施完善, 健身房专业",
  "record_id": "record-zGbVz****",
  "score": 0.51946723,
  "strategy_name": "Factual_Experience",
```

```
"updated_at": 1757907022
},
{
  "created_at": 1757907028,
  "event_ids": "[\\"event-y4YD****\\"]",
  "record_content": "用户喜欢人少、安静的地方，对美食特别感兴趣。",
  "record_id": "record-kscmC****",
  "score": 0.48046106,
  "strategy_name": "Persona_Profile",
  "updated_at": 1757907028
},
{
  "created_at": 1757907032,
  "event_ids": "[\\"event-y4YD****\\"]",
  "record_content": "用户预订酒店时需要配备健身房。",
  "record_id": "record-ALokbk****",
  "score": 0.4734102,
  "strategy_name": "Persona_Profile",
  "updated_at": 1757907032
},
{
  "created_at": 1757907019,
  "event_ids": "[\\"event-y4YD****\\"]",
  "record_content": "用户的女朋友喜欢现代艺术，故行程中也加入了现代艺术展览的
点。",
  "record_id": "record-61Mqo****",
  "score": 0.47263998,
  "strategy_name": "Factual_Experience",
  "updated_at": 1757907019
},
{
  "created_at": 1757907031,
  "event_ids": "[\\"event-y4YD****\\"]",
  "record_content": "用户的女朋友更喜欢现代艺术。",
  "record_id": "record-E0klI****",
  "score": 0.40267542,
  "strategy_name": "Persona_Profile",
  "updated_at": 1757907031
},
{
  "created_at": 1757907030,
  "event_ids": "[\\"event-y4YD****\\"]",
  "record_content": "用户对花生严重过敏，请在预订餐厅时特别注意。",
```

```
    "record_id": "record-pJ_g****",  
    "score": 0.38543546,  
    "strategy_name": "Persona_Profile",  
    "updated_at": 1757907030  
  }  
],  
  "total_count": 10  
}
```

控制台指南

新建 Memory

最近更新时间：2025-09-15 14:54:21

操作场景

创建 Memory 实质上是为您的应用初始化一个基于对话的专属数据库，便于将对话中的碎片化信息转化为可查询、可关联的结构化数据，是解锁 AI 长期记忆能力的首要操作。

ⓘ 说明：

每个账户最多可创建5个 Memory，建议您根据不同的业务或场景（如“智能客服”、“旅游咨询”）来规划 Memory 资源。

前提条件

已注册腾讯云账号并完成实名认证。

- 如需注册腾讯云账号：请单击 [注册腾讯云账号](#)。
- 如需完成实名认证：请单击 [实名认证](#)。

操作步骤

1. 使用腾讯云账号登录 [Memory 智能体记忆](#) 控制台页面。
2. 单击**新建 Memory**，进入 Memory 配置页面。
3. 参考如下表的信息，配置 Memory 的基本信息与记忆策略。

←
新建 Memory
白 免费体验时长: 3个月

基本信息

地域 ▼

广州
▼

当前仅开放广州地域，其他地域敬请期待或联系商务

服务名称 •

请输入记忆体名称，如'智能客服记忆系统'

支持 60 个字符内的中英文、数字、中划线及下划线

描述

用于描述记忆体的用途、场景或背景

0 / 200

支持 200 个字符内的中英文、数字、特殊字符：_ - + & = ! @ # \$ % ^ * ()

关联大模型

通用大模型

限时 Token 免费体验

使用大模型并搭配默认或自定义的提示词来实现长期记忆的提取

标签

标签键
▼

标签值
▼

✕

+ 添加 🔗 键值粘贴板

如现有的标签不合适，您可以去控制台[新建标签/标签值](#)

记忆配置

短期记忆容量 •

-

100
+

万条

👤 **人物画像**

通过大模型提取原文对话中的人物身份、社交关系、行为特征与个性偏好等信息，并整合成语义化的用户画像。

个性化推荐
用户画像构建
智能助理个性调优

📋 **任务信息**

通过大模型提取会话级的任务概览与进度信息，并进行会话级的持续更新，实现任务上下文的有效压缩，为长任务提供有...

任务上下文压缩
长任务执行支持
多任务协同调度

📖 **事实经验**

通过大模型提取原文对话中的经历事件、知识见解、操作流程与经验总结等信息，并整合成语义化的事实记忆。

知识库沉淀
经验复用
培训与决策支持

[+ 自定义策略](#) (还可以添加 3 个)

界面区域	界面参数	解释	限制与建议
基本信息	地域	服务数据存储的物理位置。	目前仅开放广州地域。
	服务名称	记忆体的唯一标识名称，用于管理和区分不同的记忆体服务。	<ul style="list-style-type: none"> 限制：60个字符以内，支持中英文、数字及下划线。 建议：名称应能直接反映该记忆体的用途，例如 客服工单记忆库、电商导购记忆体等。
	描述	对记忆体的使用场景、用途或者背景进行描述，便于识别与理解。	支持 200 个字符内的中英文、数字、特殊字符：_ - + & = ! @ # \$ % ^ * ()
	关联大模型	指定该记忆体服务关联的大模型。模型决定了信息处理（如总结、提取）的质量和能。	-

	标签	以键值对的形式为资源添加标识，用于项目管理、环境区分等。	-
记忆配置	短期记忆容量	<p>定义记忆体能够存储的原始对话或记忆条目的数量上限。</p> <div style="border: 1px solid #00aaff; padding: 5px; margin-top: 10px;"> <p>说明： “短期”是一个相对概念，指的是直接存储未经高度压缩的交互记录。</p> </div>	为保证系统初期的稳定性与性能，现阶段体验期内，每个 Memory 实例最多支持100万条数据。
	长期记忆策略	指定长期记忆策略，以决定记忆体如何从原始会话中提取或整合信息。	<ul style="list-style-type: none"> 默认支持人物画像、任务信息、事实经验策略。 至少选择一个策略。 单击+，可自定义添加3条策略。具体操作请见步骤4。

4. (可选) 单击**长期记忆策略**的**+**，可自定义添加新的记忆策略。



4.1 在**请输入策略标题**的输入框编辑标题，便于识别即可。

4.2 在**描述**的输入框，补充说明策略的具体用途、适用场景或背景。

4.3 在**记忆提取 Prompt** 区域的输入框，可设置信息提取的指令，单击 **Prompt 示例**，可参考示例的格式设置提示词。

说明：
记忆提取提示词是整个策略的核心指令，负责引导 AI 从单次对话或单个文本片段中初步识别并抽取出具有保存价值的信息。

4.4 在记忆整合 Prompt 区域的输入框，可设置信息整合的提示词，单击 Prompt 示例，可参考示例的格式设置提示词。

说明：
 记忆整合提示词也是整个策略的核心指令，负责引导 AI 如何将新提取的信息与记忆中已有的旧信息进行合并、去重和更新。

4.5 单击确定，完成记忆策略配置。

长期记忆策略

- 人物画像**
 通过大模型提取原文对话中的人物身份、社交关系、行为特征与个性偏好等信息，并整合...
 个性化推荐 用户画像构建 智能助理个性调优
- 任务信息**
 通过大模型提取会话级的任务概览与进度信息，并进行会话级的持续更新，实现任务上...
 任务上下文压缩 长任务执行支持 多任务协同调度
- 事实经验**
 通过大模型提取原文对话中的经历事件、知识见解、操作流程与经验总结等信息，并整合...
 知识库沉淀 经验复用 培训与决策支持

请输入策略标题

描述
 用于描述该策略的用途、场景或背景

记忆提取 Prompt Prompt 示例

你是一名长期记忆提取 Agent，负责从用户对话中提取语义信息。你的目标是提取事实性信息、实体、关系和关键概念，并整理成便于语义检索的语义句子。

<指令>

1. 仅分析用户消息，助手消息仅作参考，不直接提取。
2. 提取显性事实和可以合理推断的隐性事实。
3. 避免重复与无关内容，若无有效信息，请返回空列表。
4. 不得引入外部知识，必须基于用户输入。
5. 将提取的信息转换为完整的语义句子，便于向量化存储和检索。

输出格式请按以下格式输出语义提取结果，每个语义句子应该具有独立性和完整性：
 - [主体]在[时间/地点]进行了[具体行为/活动]

记忆整合 Prompt Prompt 示例

你是一名专业的语义记忆整合 Agent，负责将新提取的语义信息与已存在的语义记忆进行智能整合。

你的目标是将新的事实性信息与现有语义记忆进行合并、更新和去重，生成统一的语义记忆库。

<指令>

1. 分析新提取的语义信息，识别事实性信息、实体、关系和关键概念。
2. 与已存在的语义记忆进行对比，识别相似、重复或互补的语义信息。
3. 对于相似事实，进行合并和强化，保留最完整和最新的信息。
4. 对于重复事实，进行去重处理，保留最准确的版本。
5. 对于新事实，作为独立记忆添加。
6. 保持语义信息的完整性和准确性，不得引入外部知识。

整合规则

- 合并规则：相似事实合并为一条记录，保留最完整的描述
- 去重规则：完全重复的事实只保留一条
- 新增规则：全新事实作为新记录添加

取消 确定

+ 自定义策略 (还可以添加 2 个)

5. 配置完成，单击立即试用，等待任务执行完成，可在 Memory 智能体控制台 页面，查看到已创建的 Memory 卡片，如下图所示。更多信息，请参见 查看 Memory。

Memory 智能体记忆

提供智能体存储、检索和调用历史信息及上下文理解，让 Agent 实现连贯的多轮对话，并提供个性化服务。

免费试用

+ 新建 Memory



Travel_Consultation 预览版

● 运行中

ID:tdai-m

广州

通用大模型

2025-09-11 14:39 创建

详情



Initial_Test 预览版

● 运行中

ID:tdai-i

广州

通用大模型

2025-09-11 11:55 创建

详情

查看 Memory

最近更新时间：2025-09-12 21:41:51

操作场景

前往 Memory 管理控制台，即可集中查看实例的核心配置与运行状态，包括基本信息、记忆策略和访问地址等。

操作步骤

1. 使用腾讯云账号登录 [Memory 智能体记忆](#) 控制台。



2. 在每一个 Memory 卡片上，您可以查看到 Memory 名称、ID、关联的大模型、运行状态及创建时间。

说明：

Memory ID 由系统随机分配，具有唯一性，将鼠标悬停在 ID 上，点击 ID 可一键复制 Memory ID。

3. 在右上角的搜索框，可根据**服务名称**、**服务 ID**、**服务状态**、**标签**搜索需查看的 Memory。
4. 单击 Memory 卡片，进入查看 Memory 详情页面，如下所示。
 - 在**服务信息**区域，可获取 Memory 的版本信息、关联模型、创建时间及标签。
 - 在**长期记忆策略**区域，可获取长期记忆的提取策略。
 - 在**API 接入**区域，可获取 Memory 的访问地址与访问密钥。单击**接入 SDK**的**完整调用指南**可跳转至 SDK 接口文档，可使用 SDK 接口管理 Memory 的 Session、Event、State、Record 等资源。

←

Initial_Test ● 运行中

ID: tdai-mi | 广州 | --

服务概览

服务信息

版本	预览版	关联模型	通用大模型
创建时间	2025-09-11 11:55:17	标签	0 ✎

长期记忆策略

<p>人物画像 系统策略</p> <p>通过大模型提取原文对话中的人物身份、社交关系、行为特征与个性偏好等信...</p>	<p>任务信息 系统策略</p> <p>通过大模型提取会话级的任务概览与进度信息，并进行会话级的持续更新，实...</p>	<p>事实经验 系统策略</p> <p>通过大模型提取原文对话中的经历事件、知识见解、操作流程与经验总结等信...</p>
--	--	--

API 接入

- **获取 API KEY**

API Key 是服务访问的重要凭证，长期有效。请妥善保管并定期更换密钥，避免公开共享，以防安全风险和资金损失

访问地址（内网）	https://me[redacted]n.com 📄
访问密钥	***** 📄
- **接入 SDK**

复制示例代码，粘贴至业务端。[完整调用指南](#)

5.（可选）单击右上角的**编辑**，可修改 Memory 的名称。

6.（可选）单击右上角的**API 调用指南**，跳转至 API 接口文档，可使用 HTTP API 接口管理 Memory 的 Session、Event、State、Record 等资源。

开发者工具

Python SDK

创建 Client

最近更新时间：2025-09-12 21:41:51

MemoryClient() 用于通过 HTTP I/O 请求方式创建一个 Memory 的客户端对象。

```
class MemoryClient:
    def __init__(self,
                 endpoint: str,
                 api_key: str,
                 memory_id: str,
                 default_actor_id: Optional[str] = None,
                 timeout: int = 10,
                 stub: Optional[Stub] = None):
        self.default_memory_id = memory_id
        self.default_actor_id = default_actor_id
        self.stub = stub or HttpStub(endpoint, api_key, memory_id,
                                     timeout)
```

使用示例

```
from tdaimemory import MemoryClient

EndPoint = "https://memory.tdai.tencentyun.com"
ApiKey = "*****"
MemoryId = "tdai-mem-*****"
client = MemoryClient(endpoint=EndPoint,
                      api_key=ApiKey,
                      memory_id=MemoryId,
                      default_actor_id="user-sdk-test")
```

参数名	参数含义	是否必须	获取方式
endpoint	客户端所需连接 Memory 服务端的访问	是	登录 Memory 智能体控制台 ，在详情页面的 API 接入区域，可获取 DeepSearch 的访问地址与访

	地址。		问密钥。
api_key	客户端访问 Memory 服务端的 API 密钥，用于进行身份认证。	是	
memory_id	Memory 服务端具有唯一标识的 ID。	是	登录 Memory 智能体控制台 ，在 Memory 的卡片上或详情页面，可复制 ID。
timeout	请求超时时间。	否	<ul style="list-style-type: none"> • 单位：秒。 • 默认值：10。 • 取值范围：大于等于0。若 timeout 设置为小于0或为 null，系统会自动赋值为默认值。

Session 管理

创建 Session

最近更新时间：2025-09-12 21:41:51

接口定义

本接口 `create_session()` 用于创建一个新会话，并将其关联到指定的记忆库。

```
def create_session(
    name: str | None = None,
    memory_id: str | None = None,
    actor_id: str | None = None
) -> Dict[str, Any]
```

使用示例

```
import json
from tdaimemory import MemoryClient

session = client.create_session(
    memory_id="tdai-mem-e3f7****",
    actor_id="user-sdk-test",
    name="Travel_Consultation",
)
session_id = session['session_id']
print(json.dumps(session_id, ensure_ascii=False))
```

入参描述

参数	是否必选	参数含义	配置方法及要求
memory_id	否	指定 Session 需关联的记忆库 ID。	登录 Memory 智能体控制台 ，可在 Memory 服务概览页面复制 Memory ID。具体可以参见 查看 Memory 。
actor_id	否	指定 Session 的用户 ID。	<ul style="list-style-type: none">数据类型：String长度限制：[1,20]字符规则：任意字符

name	是	指定 Session 名称。	<ul style="list-style-type: none">数据类型: String长度限制: [1,128]字符规则: 任意字符
------	---	----------------	---

出参描述

执行成功，输出如下信息。

```
Session created successfully: session-1WNJ****
```

参数名	参数含义
session_id	为所创建的 Session 自动分配的具有唯一标识的 ID。

查询 Session

最近更新时间：2025-09-12 21:41:51

接口定义

本接口 `query_sessions()` 用于获取与特定 Memory 和用户关联的所有 Session 列表。

```
def query_sessions(
    memory_id: str | None = None,
    actor_id: str | None = None,
    order_by: Dict[str, int] | None = None,
    limit: int | None = None,
    offset: int | None = None
) -> Dict[str, Any]
```

使用示例

```
import json
from tdaimemory import MemoryClient

sessions = client.query_sessions(
    order_by={"created_at": -1},
    limit=3,
    offset=0
)
print(json.dumps(sessions, ensure_ascii=False))
```

入参描述

参数	是否必选	参数含义	配置方法及要求
<code>memory_id</code>	否	指定查询 Session 需关联的记忆库 ID。	登录 Memory 智能体控制台 ，可在 Memory 服务概览页面复制 Memory ID。具体可以参见 查看 Memory 。
<code>actor_id</code>	否	指定查询 Session 的用户 ID。	<ul style="list-style-type: none">数据类型：String长度限制：[1,20]字符规则：任意字符

order_by	否	指定返回的 Session 的排序方式。	支持如下两种方式。 <ul style="list-style-type: none"> • -1: 降序。默认按照 Session 的创建时间降序排列。 • 1: 升序。
limit	否	指定返回的会话数量。	<ul style="list-style-type: none"> • 数据类型: int, 正整数。 • 取值范围: 1~100。 • 默认值: 10。
offset	否	指定返回结果列表的起始偏移量。	<ul style="list-style-type: none"> • 数据类型: int, 正整数。 • 默认值: 0, 表示从数据集的第一条记录开始返回。 • 示例: offset=20 表示跳过前20条记录, 从第21条开始返回。

出参描述

```

{
  "sessions": [
    {
      "created_at": 1757407397,
      "session_id": "session-1WNJ****",
      "session_name": "Travel_Consultation",
      "updated_at": 1757407397
    }
  ],
  "total_count": 1
}

```

参数名	参数含义
seesions	列出满足查询条件的所有 Session。 <ul style="list-style-type: none"> • session_id: Session 的唯一标识 ID。 • session_name: Session 的会话名称。 • created_at: Session 的创建时间。 • updated_at: Session 的最后一次更新时间。
total_count	满足查询条件的 Session 数量。

更新 Session

最近更新时间：2025-09-12 21:41:51

接口定义

本接口 `update_session()` 用于更新一个已存在 Session 的属性信息，例如，更新名称。

```
def update_session(
    session_id: str,
    name: str,
    memory_id: str | None = None,
    actor_id: str | None = None
) -> Dict[str, Any]
```

使用示例

```
import json
from tdaimemory import MemoryClient

updated = client.update_session(
    session_id="session-Y5oi****",
    name="Travel_Consultation_update",
)
print(json.dumps(updated, ensure_ascii=False))
```

入参描述

参数	是否必选	参数含义	配置方法及要求
memory_id	否	指定所更新 Session 所关联的 Memory ID。	登录 Memory 智能体控制台 ，可在 Memory 服务概览页面复制 Memory ID。具体可以参见 查看 Memory 。
actor_id	否	指定所更新 Session 的用户 ID。	<ul style="list-style-type: none">数据类型：String长度限制：[1,20]字符规则：任意字符
session_id	是	指定所需要更新的 Session ID。	获取 ID，请参见 查询 Session 。
name	是	指定更新后的 Session 名称。	<ul style="list-style-type: none">数据类型：String

- 长度限制: [1,128]
- 字符规则: 任意字符

出参描述

执行成功，输出如下信息。

```
>>> print(json.dumps(updated, ensure_ascii=False))
{"affected_count": 1}
```

参数名	参数含义
affected_count	返回变更 Session 的数量。

删除 Session

最近更新时间：2025-09-12 21:41:51

接口定义

本接口 `delete_session()` 用于永久删除一个指定的会话（Session）及其相关数据。

```
def delete_session(
    session_id: str,
    memory_id: str | None = None,
    actor_id: str | None = None
) -> Dict[str, Any]
```

使用示例

```
import json
from tdaimemory import MemoryClient

delete_result = client.delete_session(
    session_id="session-Y5oi**",
)
print(json.dumps(delete_result, ensure_ascii=False))
```

请求参数

参数	是否必选	参数含义	配置方法及要求
memory_id	否	指定所需删除 Session 所关联的 Memory ID。	登录 Memory 智能体控制台 ，可在 Memory 服务概览页面复制 Memory ID。具体可以参见 查看 Memory 。
actor_id	否	指定删除 Session 的用户 ID。	<ul style="list-style-type: none">数据类型：String长度限制：[1,20]字符规则：任意字符
session_id	是	指定所需删除的 Session ID。	获取 ID，请参见 查询 Session 。

响应消息

执行成功，输出如下信息。

```
>>> print(json.dumps(delete_result, ensure_ascii=False))  
{"affected_count": 1}
```

参数名	参数含义
affected_count	返回删除 Session 的数量。

搜索 Session

最近更新时间：2025-09-12 21:41:51

接口定义

本接口 `search_sessions()` 提供基于关键词规则匹配的会话搜索功能，可根据会话名称检索并返回符合条件的会话列表。

```
def search_sessions(
    content: str,
    memory_id: str | None = None,
    actor_id: str | None = None,
    order_by: Dict[str, int] | None = None,
    limit: int | None = None
) -> Dict[str, Any]
```

使用示例

```
import json
from tdaimemory import MemoryClient

search_result = client.search_sessions(
    content="Travel",
    order_by={"created_at": -1},
    limit=2
)
print(json.dumps(search_result, ensure_ascii=False))
```

入参描述

参数	是否必选	参数含义	配置方法及要求
<code>memory_id</code>	否	指定查询 Session 需关联的记忆库 ID。	登录 Memory 智能体控制台 ，可在 Memory 服务概览页面复制 Memory ID。具体可以参见 查看 Memory 。
<code>actor_id</code>	否	指定查询 Session 的用户 ID。	<ul style="list-style-type: none">数据类型：String长度限制：[1,20]字符规则：任意字符

content	是	指定搜索的关键词。	数据类型: String。 限制: 当前仅支持使用 Session Name 中包含的关键词搜索。
order_by	否	指定返回的 Session 的排序方式。	支持如下两种方式。 <ul style="list-style-type: none"> • -1: 降序。默认按照 Session 的创建时间降序排列。 • 1: 升序。
limit	否	返回的最大会话数量。	<ul style="list-style-type: none"> • 数据类型: int。 • 默认值: 3 • 取值范围: [1,50]。

响应消息

执行成功，输出如下信息。

```
>>> print(json.dumps(search_result, ensure_ascii=False))
{"sessions": [{"created_at": 1757579185, "session_id": "session-KaxA****", "session_name": "Travel_Consultation_update", "updated_at": 1757579197}], "total_count": 2}
```

参数名	参数含义
seesions	列出满足查询条件的所有 Session。 <ul style="list-style-type: none"> • session_id: Session 的唯一标识 ID。 • session_name: Session 的会话名称。 • created_at: Session 的创建时间。 • updated_at: Session 的最后一次更新时间。
total_count	满足查询条件的 Session 数量。

Event 管理

追加 Event

最近更新时间：2025-09-12 21:41:51

接口定义

本接口 `append_event()` 用于向指定会话（Session）中添加新的对话事件。

```
def append_event(  
    session_id: str,  
    messages: dict,  
    memory_id: str | None = None,  
    actor_id: str | None = None  
) -> Dict[str, Any]
```

使用示例

```
import json  
from tdaimemory import MemoryClient  
  
event = client.append_event(  
    session_id="session-Y5oi****",  
    messages={  
        "chat": [  
            {  
                "role": "user",  
                "content": "嗨，帮我规划一个下周去杭州的3天行程。"  
            },  
            {  
                "role": "assistant",  
                "content": "好的，已为您推荐杭州三日行程。第一天，我们聚焦西湖经典。清晨从断桥残雪出发，漫步白堤，感受水墨画般的湖光山色。随后乘船登岛探访三潭印月，下午登雷峰塔俯瞰全景，傍晚在曲院风荷赏夕阳，晚上则去河坊街夜市品尝地道小吃。\\n\\n第三天，探索湿地与运河文化。上午在西溪湿地乘坐摇橹船，听船娘小调，感受水乡慢生活；下午逛一逛桥西历史街区和文艺的小河直街，参观工艺美术博物馆，在咖啡馆小憩后即可轻松结束旅程，准备返程。\\n\\n第二天，体验禅意与宋韵。上午前往灵隐寺景区祈福，在法喜寺的黄墙拍照打卡；下午沉浸于宋城，体验AR剧本杀并观看《宋城千古情》演出；如果傍晚还有精力，可以去九溪烟树享受清凉的溪水。"  
            }  
        ]  
    },  
    memory_id="memory-123456789",  
    actor_id="actor-123456789")
```

```
{
  "role": "user",
  "content": "我喜欢人少、安静的地方，对美食特别感兴趣。",
},
{
  "role": "assistant",
  "content": "基于您喜欢安静和美食的偏好，我为您优化了行程，避开了团队游客扎堆的景点，专注于幽静的体验和地道的风味。\\n\\n第一天：避世西湖\\n清晨漫步杨公堤与茅家埠，享受静谧湖景。午餐在老字号「奎元馆」吃虾爆鳝面。下午乘船访三潭印月，傍晚在湖边茶社小坐。晚餐去本地人爱的胜利河美食街尝地道风味。\\n\\n第二天：禅意之路\\n一早探灵隐寺，然后沿幽静天竺路步行至法喜寺。午餐在「知竹」吃素面。下午深入云栖竹径看竹海。晚餐用「新丰小吃」的喉口馒头等市井味道收尾。\\n\\n第三天：水乡慢生活\\n全天在西溪湿地坐摇橹船，深入芦苇荡享受宁静。午后往文艺的小河直街，在运河畔的咖啡馆小坐休憩，结束旅程。",
},
{
  "role": "user",
  "content": "太好了，酒店要带健身房，另外我对花生严重过敏，预订餐厅时请务必注意。",
},
{
  "role": "assistant",
  "content": "好的，已完全了解您的需求（偏好安静人少、喜爱美食、酒店需带健身房、对花生严重过敏）。为您精选的酒店（带健身房）：\\n\\n西湖庐驿精品酒店：位于西湖区，设计静谧，配备健身房，且远离闹市，符合您对安静的需求。\\n\\n杭州西溪紫金港希尔顿逸林酒店：靠近西溪湿地，设施完善，健身房专业，适合在第三天安排湿地行程时入住。",
},
{
  "role": "user",
  "content": "完美！哦对了，我女朋友更喜欢现代艺术，有什么地方可以推荐吗？",
},
{
  "role": "assistant",
  "content": "收到！考虑到您女朋友的喜好。这是为您和您女友定制的杭州三日行程：\\n\\n第一天：上午游茅家埠、杨公堤；下午观天目里美术馆现代展览。晚餐于胜利河美食街 品尝安全地道的本帮菜。\\n\\n第二天：上午早访灵隐寺，漫步幽静天竺路。下午重点参观浙江省博物馆之江馆 重磅国际艺术大展。晚餐体验本地 新丰小吃（已规避风险）。\\n\\n第三天：上午于 西溪湿地乘摇橹船。下午在小河直街 咖啡馆悠闲小憩，结束旅程。",
}
],
}
)
```

```
print(json.dumps(event, ensure_ascii=False))
```

请求参数

参数	是否必选	参数含义	配置方法及要求
memory_id	否	指定追加事件的会话 (Session) 所关联的 Memory ID。	登录 Memory 智能体控制台 ，可在 Memory 服务概览页面复制 Memory ID。具体可以参见 查看 Memory 。
actor_id	否	指定追加事件的会话 (Session) 中所属用户的唯一标识 ID。	<ul style="list-style-type: none"> 数据类型: String 长度限制: [1,20] 字符规则: 任意字符
session_id	是	指定追加 Event 的 Session ID。	获取会话 ID，请参见 查询 Session 。
messages	是	指定追加事件的消息对象。	<p>以 json 格式配置消息体。本示例以 role (角色) 和 content (内容) 两个字段，记录事件中不同参与者 (如用户、助手、工具、系统) 的具体交互内容。</p> <ul style="list-style-type: none"> role: 指定发言者身份。 <ul style="list-style-type: none"> user: 表示最终用户。 assistant: AI 助手。 tool: 被调用的工具或函数。 system: 系统指令或通知。 content: 对应角色发出的实际文本内容或数据。

响应消息

执行成功，输出如下信息。

```
>>> print(json.dumps(event, ensure_ascii=False))
{"event_id": "event-Ih-*****"}
```

参数名	参数含义
event_id	返回创建的事件 ID。

查询 Event

最近更新时间：2025-09-12 21:41:52

接口定义

本接口 `query_events()` 用于查询会话中的事件。

```
def query_events(
    memory_id: str | None = None,
    actor_id: str | None = None,
    session_id: str | None = None,
    where: Dict[str, Any] | None = None,
    order_by: Dict[str, int] | None = None,
    limit: int | None = None,
    offset: int | None = None
) -> Dict[str, Any]
```

使用示例

```
import json
from tdaimemory import MemoryClient

events = client.query_events(
    session_id="session-hV1W****",
    # where={"type": "message"},
    order_by={"created_at": -1},
    limit=20,
    offset=0
)
print(json.dumps(events, ensure_ascii=False))
```

请求参数

参数	是否必选	参数含义	配置方法及要求
<code>memory_id</code>	否	指定查询会话（Session）所关联的 Memory ID。	登录 Memory 智能体控制台 ，可在 Memory 服务概览页面复制 Memory ID。具体可以参见 查看 Memory 。

actor_id	否	指定查询会话 (Session) 中所属用户的唯一标识 ID。	<ul style="list-style-type: none"> 数据类型: String 长度限制: [1,20] 字符规则: 任意字符
session_id	是	指定查询事件的 Session ID。	获取会话 ID, 请参见 查询 Session 。
where	否	where 指定查询条件, 返回完全符合这些规则的事件。	<p>设置规则为: 以键值对的形式, 定义事件消息体中字段的匹配规则。</p> <ul style="list-style-type: none"> 支持使用的操作符。 <ul style="list-style-type: none"> 比较操作符: \$eq (等于, 默认可省略)、\$ne (不等于)、\$gt (大于)、\$gte (大于等于)、\$lt (小于)、\$lte (小于等于)。 集合操作符: \$in (包含在数组中)、\$nin (不包含在数组中) 如下表示查找 status 字段等于 "ACTIVE" 且 score 字段大于等于80的记录。 <pre data-bbox="874 1003 1481 1375"> { "where": { "status": "ACTIVE", "score": { "\$gte": 80 } } } </pre>
order_by	否	指定返回的 Event 的排序方式。	<p>支持如下两种方式。</p> <ul style="list-style-type: none"> -1: 降序。默认按照 Event 的创建时间降序排列。 1: 升序。
limit	否	指定返回的事件数量。	<ul style="list-style-type: none"> 数据类型: int, 正整数。 取值范围: 1~100。 默认值: 10。
offset	否	指定返回结果列表的起始偏移量。	<ul style="list-style-type: none"> 数据类型: int, 正整数。 默认值: 0, 表示从数据集的第一条记录开始返回。

- 示例：offset=20 表示跳过前20条记录，从第21条开始返回。

响应消息

执行成功，输出如下信息。

```
>>> print(json.dumps(events, ensure_ascii=False, indent=6))
{
  "events": [
    {
      "chat": [
        {
          "content": "嗨，帮我规划一个下周去杭州的3天行程。",
          "role": "user"
        },
        {
          "content": "好的，已为您推荐杭州三日行程。第一天，我们聚焦西湖经典。清晨从断桥残雪出发，漫步白堤，感受水墨画般的湖光山色。随后乘船登岛探访三潭印月，下午登雷峰塔俯瞰全景，傍晚在曲院风荷赏夕阳，晚上则去河坊街夜市品尝地道小吃。\\n\\n第三天，探索湿地与运河文化。上午在西溪湿地乘坐摇橹船，听船娘小调，感受水乡慢生活；下午逛一逛桥西历史街区和文艺的小河直街，参观工艺美术博物馆，在咖啡馆小憩后即可轻松结束旅程，准备返程。\\n\\n第二天，体验禅意与宋韵。上午前往灵隐寺景区祈福，在法喜寺的黄墙拍照打卡；下午沉浸于宋城，体验AR剧本杀并观看《宋城千古情》演出；如果傍晚还有精力，可以去九溪烟树享受清凉的溪水。",
          "role": "assistant"
        },
        {
          "content": "我喜欢人少、安静的地方，对美食特别感兴趣。",
          "role": "user"
        },
        {
          "content": "基于您喜欢安静和美食的偏好，我为您优化了行程，避开了团队游客扎堆的景点，专注于幽静的体验和地道的风味。\\n\\n第一天：避世西湖\\n清晨漫步杨公堤与茅家埠，享受静谧湖景。午餐在老字号「奎元馆」吃虾爆鳝面。下午乘船访三潭印月，傍晚在湖边茶社小坐。晚餐去本地人爱的胜利河美食街尝地道风味。\\n\\n第二天：禅意之路\\n一早探灵隐寺，然后沿幽静天竺路步行至法喜寺。午餐在「知竹」吃素面。下午深入云栖竹径看竹海。晚餐用「新丰小吃」的喉口馒头等市井味道收尾。\\n\\n第三天：水乡慢生活\\n全天在西溪湿地坐摇橹船，深入芦苇荡享受宁静。午后往文艺的小河直街，在运河畔的咖啡馆小坐休憩，结束旅程。",
          "role": "assistant"
        }
      ]
    }
  ]
}
```

```

    },
    {
      "content": "太好了，酒店要带健身房，另外我对花生严重过敏，预订餐厅时请务必注意。",
      "role": "user"
    },
    {
      "content": "好的，已完全了解您的需求（偏好安静人少、喜爱美食、酒店需带健身房、对花生严重过敏）。为您精选的酒店（带健身房）：\n\n西湖庐驿精品酒店：位于西湖区，设计静谧，配备健身房，且远离闹市，符合您对安静的需求。\n\n杭州西溪紫金港希尔顿逸林酒店：靠近西溪湿地，设施完善，健身房专业，适合在第三天安排湿地行程时入住。",
      "role": "assistant"
    },
    {
      "content": "完美！哦对了，我女朋友更喜欢现代艺术，有什么地方可以推荐吗？",
      "role": "user"
    },
    {
      "content": "收到！考虑到您女朋友的喜好。这是为您和您女友定制的杭州三日行程：\n\n第一天：上午游茅家埠、杨公堤；下午观天目里美术馆现代展览。晚餐于胜利河美食街 品尝安全地道的本帮菜。\n\n第二天：上午早访灵隐寺，漫步幽静天竺路。下午重点参观 浙江省博物馆之江馆 重磅国际艺术大展。晚餐体验本地 新丰小吃（已规避风险）。\n\n第三天：上午于 西溪湿地乘摇橹船。下午在小河直街 咖啡馆悠闲小憩，结束旅程。",
      "role": "assistant"
    }
  ],
  "created_at": 1757412422,
  "event_id": "event-EuY_****",
  "updated_at": 1757412422
}
],
"total_count": 1
}

```

参数名 (一级)	参数名 (二级)	参数含义
events	event_id	返回满足条件的事件 ID。

	messages	返回事件的消息对象。
	created_at	事件创建时间。
total_count	-	符合条件的事件总数。

删除 Event

最近更新时间：2025-09-12 21:41:52

接口定义

本接口 `delete_event()` 用于从记忆库中永久删除指定的事件记录及相关数据。

```
def delete_event(  
    session_id: str,  
    event_id: str,  
    memory_id: str | None = None,  
    actor_id: str | None = None  
) -> Dict[str, Any]
```

使用示例

```
import json  
from tdaimemory import MemoryClient  
  
del_result = client.delete_event(  
    session_id="session-hV1W****",  
    event_id="event-sBMJ****",  
)  
print(json.dumps(del_result, ensure_ascii=False))
```

请求参数

参数	是否必选	参数含义	配置方法及要求
<code>memory_id</code>	否	指定需删除事件所属会话所关联的 Memory ID。	登录 Memory 智能体控制台 ，可在 Memory 服务概览页面复制 Memory ID。具体可以参见 查看 Memory 。
<code>actor_id</code>	否	指定需删除事件所属会话的用户 ID。	<ul style="list-style-type: none">数据类型：String长度限制：[1,20]字符规则：任意字符
<code>session_id</code>	是	指定需删除事件所属会话 ID。	获取会话 ID，请参见 查询 Session 。

event_id	是	指定需删除的具体事件 ID。	获取事件 ID，请参见 查询 Event 。
----------	---	----------------	--

响应消息

执行成功，输出如下信息。

```
>>> print(json.dumps(del_result, ensure_ascii=False, indent=7))
{
    "affected_count": 1
}
```

参数名	参数含义
affected_count	返回删除 Event 的数量。

State 管理

设置 State

最近更新时间：2025-09-12 21:41:52

接口定义

本接口 `set_state()` 用于设置当前会话中的状态数据。

```
def set_state(
    session_id: str,
    state: Dict[str, Any],
    memory_id: str | None = None,
    actor_id: str | None = None
) -> Dict[str, Any]
```

使用示例

```
import json
from tdaimemory import MemoryClient

set_state_result = client.set_state(
    session_id="session-Y5oi****",
    state={
        "user_language": "en-US",
        "notification_enabled": True,
        "环境偏好": "安静、人少",
    },
)

print(json.dumps(set_state_result, ensure_ascii=False))
```

请求参数

参数	是否必选	参数含义	配置方法及要求
memory_id	否	指定需设置当前会话状态的 Memory ID。	登录 Memory 智能体控制台 ，可在 Memory 服务概览页面复制 Memory ID。具体可以参见 查看 Memory 。

actor_id	否	指定需设置当前会话状态的用户 ID。	<ul style="list-style-type: none"> 数据类型: String 长度限制: [1,20] 字符规则: 任意字符
session_id	是	指定需设置当前会话状态的会话 ID。	获取会话 ID, 请参见 查询 Session 。
state	是	指定需设置的状态数据, 对于已存在的更新, 不存在的新增。	<p>允许包含一个或多个任意类型的键值对, 如字符串、数字、布尔值、数组、对象等。示例如下所示。</p> <pre> { "user_preference_the me": "light", // 字符串值 "shopping_cart_items ": ["laptop", "headphones"], // 数 组/列表值 "user_language": "zh-CN", // 字符串值 "notification_enable d": true // 布尔值 } </pre>

响应消息

执行成功, 输出如下信息。

```
>>> print(json.dumps(set_state_result, ensure_ascii=False))
{"affected_count": 3}
```

参数名	参数含义
affected_count	设置成功的状态数量。

获取 State

最近更新时间：2025-09-12 21:41:52

接口定义

本接口 `get_states()` 用于获取当前会话中所有状态数据。

```
def get_states(
    session_id: str,
    keys: List[str] | None,
    memory_id: str | None = None,
    actor_id: str | None = None
) -> Dict[str, Any]
```

使用示例

```
import json
from tdaimemory import MemoryClient

state = client.get_states(
    session_id="session-Y5oi****",
    keys=["notification_enabled", "user_language"],
)
print(json.dumps(state, ensure_ascii=False))
```

求参数

参数	是否必选	参数含义	配置方法及要求
<code>memory_id</code>	否	指定需获取当前会话状态的 Memory ID。	登录 Memory 智能体控制台 ，可在 Memory 服务概览页面复制 Memory ID。具体可以参见 查看 Memory 。
<code>actor_id</code>	是	指定需获取当前会话状态的用户 ID。	<ul style="list-style-type: none">数据类型：String长度限制：[1,20]字符规则：任意字符
<code>session_id</code>	是	指定需获取当前会话状态的会话 ID。	获取会话 ID，请参见 查询 Session 。

keys	是	指定需获取当前会话的状态键列表。	<ul style="list-style-type: none">数据类型: String 数组示例: ["user_preference_theme", "shopping_cart_items"]
------	---	------------------	--

响应消息

执行成功，输出如下信息。

```
>>> print(json.dumps(state, ensure_ascii=False))
{"state": {"notification_enabled": true, "user_language": "en-US"}}
```

参数名	参数含义
state	返回状态数据 Key 对应的 Value。

获取所有 State

最近更新时间：2025-09-12 21:41:52

接口定义

本接口 `get_states()` 用于获取当前会话中所有状态数据。

```
def get_all_states(
    session_id: str,
    memory_id: str | None = None,
    actor_id: str | None = None
) -> Dict[str, Any]
```

使用示例

```
import json
from tdaimemory import MemoryClient

all_states = client.get_all_states(
    session_id="session-Y5oi****",
)
print(json.dumps(all_states, ensure_ascii=False))
```

请求参数

参数	是否必选	参数含义	配置方法及要求
memory_id	否	指定需获取当前会话状态的 Memory ID。	登录 Memory 智能体控制台 ，可在 Memory 服务概览页面复制 Memory ID。具体可以参见 查看 Memory 。
actor_id	否	指定需获取当前会话状态的用户 ID。	<ul style="list-style-type: none">数据类型：String长度限制：[1,20]字符规则：任意字符
session_id	是	指定需获取当前会话状态的会话 ID。	获取会话 ID，请参见 查询 Session 。

响应消息

执行成功，输出如下信息。

```
>>> print(json.dumps(all_states, ensure_ascii=False))
{"state": {"notification_enabled": true, "user_language": "en-US", "环境偏好": "安静、人少"}}
```

参数名	参数含义
state	返回所有状态数据。

删除 State

最近更新时间：2025-09-12 21:41:52

接口定义

本接口 `delete_states()` 用于删除会话中指定的一个或多个状态键。

```
def delete_states(
    session_id: str,
    keys: List[str] | None,
    memory_id: str | None = None,
    actor_id: str | None = None
) -> Dict[str, Any]
```

使用示例

```
import json
from tdaimemory import MemoryClient

del_state_result = client.delete_states(
    session_id="session-Y5oi****",
    keys=["notification_enabled"],
)

print(json.dumps(del_state_result, ensure_ascii=False))
```

请求参数

参数	是否必选	参数含义	配置方法及要求
<code>memory_id</code>	否	指定需删除会话状态的 Memory ID。	登录 Memory 智能体控制台 ，可在 Memory 服务概览页面复制 Memory ID。具体可以参见 查看 Memory 。
<code>actor_id</code>	否	指定需删除会话状态的用户 ID。	<ul style="list-style-type: none">数据类型：String长度限制：[1,20]字符规则：任意字符
<code>session</code>	是	指定需删除会话状态的会	获取会话 ID，请参见 查询 Session 。

_id		话 ID。	
keys	是	指定需删除的状态键列表。	<ul style="list-style-type: none">数据类型：String 数组。示例：["user_preference_theme", "shopping_cart_items"]

响应消息

执行成功，输出如下信息。

```
>>> print(json.dumps(del_state_result, ensure_ascii=False))
{"affected_count": 1}
```

参数名	参数含义
affected_count	返回删除的状态键数量。

清空 State

最近更新时间：2025-09-12 21:41:52

接口定义

本接口 `flush_states()` 用于清空会话中的所有状态数据。

⚠ 注意：

本接口功能为删除当前会话中的全部状态键值对。此操作执行后数据将不可恢复，请您务必谨慎确认！

```
def flush_states(  
    session_id: str,  
    memory_id: str | None = None,  
    actor_id: str | None = None  
) -> Dict[str, Any]
```

使用示例

```
import json  
from tdaimemory import MemoryClient  
  
flush_result = client.flush_states(  
    session_id="session-Y5oi****",  
)  
print(json.dumps(flush_result, ensure_ascii=False))
```

请求参数

参数	是否必选	参数含义	配置方法及要求
<code>memory_id</code>	否	指定需清空会话状态的 Memory ID。	登录 Memory 智能体控制台 ，可在 Memory 服务概览页面复制 Memory ID。具体可以参见 查看 Memory 。
<code>actor_id</code>	是	指定需清空会话状态的用户 ID。	<ul style="list-style-type: none">数据类型：String长度限制：[1,20]字符规则：任意字符

<code>session_id</code>	是	指定需清空会话状态的会话 ID。	获取会话 ID，请参见 查询 Session 。
-------------------------	---	------------------	--

响应消息

执行成功，输出如下信息。

```
>>> print(json.dumps(flush_result, ensure_ascii=False))
{"affected_count": 2}
```

参数名	参数含义
<code>affected_count</code>	返回清空的状态键数量。

Record 管理

创建 Record

最近更新时间：2025-09-12 21:41:52

接口定义

本接口 `append_record()` 用于创建新的长期记忆记录，用于存储用户自定义的个性化信息，如用户偏好、重要事实等。

```
def append_record(
    session_id: str,
    content: str,
    strategy: str,
    memory_id: str | None = None,
    actor_id: str | None = None
) -> Dict[str, Any]
```

使用示例

```
import json
from tdaimemory import MemoryClient

record = client.append_record(
    session_id="session-Y5oi****",
    content="美食维度为地道风味优先",
    strategy="旅游关注维度",
)
print(json.dumps(record, ensure_ascii=False))
```

请求参数

参数	是否必选	参数含义	配置方法及要求
<code>memory_id</code>	否	指定需创建长期记忆的 Memory ID。	登录 Memory 智能体控制台 ，可在 Memory 服务概览页面复制 Memory ID。具体可以参见 查看 Memory 。

actor_id	否	指定需创建长期记忆的用户 ID。	<ul style="list-style-type: none"> 数据类型: String 长度限制: [1,20] 字符规则: 任意字符
session_id	是	指定需创建长期记忆的会话 ID。	获取会话 ID, 请参见 查询 Session 。
content	是	记录内容, 描述记忆的实际信息。	<ul style="list-style-type: none"> 数据类型: 字符串 示例: "用户偏好使用深色主题, 喜欢简洁的界面设计"。
strategy	是	指定长期记忆的策略类型。	可设置的策略类型, 默认支持如下策略, 支持自定义其他策略: <ul style="list-style-type: none"> Persona_Profile: 人物画像。 Task_Information: 任务信息。 Factual_Experience: 事实经验。

响应消息

执行成功, 输出如下信息。

```
>>> print(json.dumps(record, ensure_ascii=False))
{"record_id": "record-Un4a****"}
```

参数名	参数含义
record_id	系统随机分配的长期记忆 ID, 用于唯一标识该记录。

查询 Record

最近更新时间：2025-09-12 21:41:52

接口定义

本接口 `query_records()` 用于查询指定条件下的长期记忆。

- 支持根据 Memory ID、Actor ID、Seesion ID 等条件筛选 Record。
- 支持分页，便于管理和浏览记录数据。

```
def query_records(
    memory_id: str | None = None,
    actor_id: str | None = None,
    session_id: str | None = None,
    strategies: List[str] | None = None,
    where: Dict[str, Any] | None = None,
    order_by: Dict[str, int] | None = None,
    limit: int | None = None,
    offset: int | None = None
) -> Dict[str, Any]
```

使用示例

```
import json
from tdaimemory import MemoryClient

records = client.query_records(
    session_id="session-Y5oi****",
    #strategies=["CONSULT_STRATEGY"],
    # where={"importance": "high"},
    order_by={"created_at": -1},
    limit=15,
    offset=0
)
print(json.dumps(records, ensure_ascii=False))
```

请求参数

参数	是否必选	参数含义	配置方法及要求
----	------	------	---------

memory_id	否	指定查询 Record 所关联的 Memory ID。	登录 Memory 智能体控制台 ，可在 Memory 服务概览页面复制 Memory ID。具体可以参见 查看 Memory 。
actor_id	否	指定查询 Record 用户的唯一标识 ID。	<ul style="list-style-type: none"> 数据类型：String 长度限制：[1,20] 字符规则：任意字符
session_id	是	指定查询 Record 的 Session ID。	获取会话 ID，请参见 查询 Session 。
strategy	否	指定 Record 策略类型。	<p>数据类型为 String 数组。默认支持如下策略，可自定义其他策略。</p> <ul style="list-style-type: none"> Persona_Profile：人物画像。 Task_Information：任务信息。 Factual_Experience：事实经验。
where	否	<p>where 指定查询条件，返回完全符合这些规则的事件。</p> <div style="border: 1px solid #00aaff; padding: 5px; margin-top: 10px;"> <p>说明： 当前仅支持使用 record_id、created_at、updated_at 字段设置查询条件。</p> </div>	<p>设置规则为：以键值对的形式，定义事件消息体中字段的匹配规则。</p> <ul style="list-style-type: none"> 支持使用的操作符。 <ul style="list-style-type: none"> 比较操作符：\$eq（等于，默认可省略）、\$ne（不等于）、\$gt（大于）、\$gte（大于等于）、\$lt（小于）、\$lte（小于等于）。 集合操作符：\$in（包含在数组中）、\$nin（不包含在数组中） 如下表示查找 status 字段等于 "ACTIVE" 且 score 字段大于等于80的记录。 <pre style="background-color: #2d3748; color: #e2e8f0; padding: 10px; border-radius: 5px;"> { "where": { "status": "ACTIVE", "score": { "\$gte": 80 } } }</pre>
order_by	否	指定返回的 Record 的排序方式。	<p>支持如下两种方式。</p> <ul style="list-style-type: none"> -1：降序。默认按照 Record 的创建时间降序排列。 1：升序。

limit	否	指定返回的 Record 数量。	<ul style="list-style-type: none"> 数据类型: int, 正整数。 取值范围: 1~100。 默认值: 10。
offset	否	指定返回结果列表的起始偏移量。	<ul style="list-style-type: none"> 数据类型: int, 正整数。 默认值: 0, 表示从数据集的第一条记录开始返回。 示例: offset=20表示跳过前20条记录, 从第21条开始返回。

响应消息

执行成功, 输出如下信息。

```
>>> print(json.dumps(records, ensure_ascii=False))
{
  "records": [
    {
      "created_at": 1757412735,
      "event_ids": "[\"event-EuY_****\"]",
      "record_content": ", 在旅游方面, [++] 喜欢人少、安静的地方",
      "record_id": "record-RJTh****",
      "strategy_name": "旅游关注维度",
      "updated_at": 1757412735
    },
    {
      "created_at": 1757412738,
      "event_ids": "[\"event-EuY_****\"]",
      "record_content": "在旅游方面, [++] 对美食特别感兴趣",
      "record_id": "record-va****rW",
      "strategy_name": "旅游关注维度",
      "updated_at": 1757412738
    },
    {
      "created_at": 1757412740,
      "event_ids": "[\"event-EuY_****\"]",
      "record_content": "在旅游方面, [++] 酒店要带健身房",
      "record_id": "record-cQyh****",
      "strategy_name": "旅游关注维度",
      "updated_at": 1757412740
    },
  ]
}
```

```

    "created_at": 1757412743,
    "event_ids": "[\"event-EuY_****\"]",
    "record_content": "在旅游方面, [---]对花生严重过敏",
    "record_id": "record-CaH7****",
    "strategy_name": "旅游关注维度",
    "updated_at": 1757412743
  },
  {
    "created_at": 1757412746,
    "event_ids": "[\"event-EuY_****\"]",
    "record_content": "在旅游方面, [++]女朋友更喜欢现代艺术",
    "record_id": "record-Wet-****",
    "strategy_name": "旅游关注维度",
    "updated_at": 1757412746
  },
  {
    "created_at": 1757414225,
    "event_ids": "[****]",
    "record_content": "美食维度为地道风味。",
    "record_id": "record-Un4a****",
    "strategy_name": "旅游关注维度",
    "updated_at": 1757414225
  }
],
"total_count": 6
}

```

参数 (一级)	参数 (二级)	参数含义
records	record_id	长期记忆 ID, 唯一标识该记录的标识符。
	record_content	记录内容, 描述 Record 记忆的实际信息。
	strategy_name	生成该记录的策略名称, 例如: User_Preference_Strategy、Semantic_Strategy 等。
	created_at	Record 创建时间。
	updated_at	Record 最后一次更新时间。

	event_ids	事件 ID。
total_count	-	符合查询条件的总 Record 数量。

更新 Record

最近更新时间：2025-09-12 21:41:52

接口定义

本接口 `update_record()` 用于更新已存在的 Record 的信息。

```
def update_record(
    session_id: str,
    record_id: str,
    content: str,
    memory_id: str | None = None,
    actor_id: str | None = None
) -> Dict[str, Any]
```

使用示例

```
import json
from tdaimemory import MemoryClient

update_result = client.update_record(
    session_id="session-Y5oi****",
    record_id="record-RJTh****",
    content="旅游环境，喜欢人少、安静的地方",
)
print(json.dumps(update_result, ensure_ascii=False))
```

请求参数

参数	是否必选	参数含义	配置方法及要求
memory_id	否	指定需更新 Record 的 Memory ID。	登录 Memory 智能体控制台 ，可在 Memory 服务概览页面复制 Memory ID。具体可以参见 查看 Memory 。
actor_id	否	指定需更新 Record 的用户 ID。	<ul style="list-style-type: none">数据类型：String长度限制：[1,20]字符规则：任意字符

session_id	是	指定需更新 Record 的会话 ID。	获取会话 ID，请参见 查询 Session 。
record_id	是	指定需更新 Record ID。	获取 Record ID，请参见 查询 Record 。
content	是	编辑更新后的记录内容。	<ul style="list-style-type: none">数据类型：String。支持输入中文、英文等任意字符。

响应消息

执行成功，输出如下信息。

```
>>> print(json.dumps(update_result, ensure_ascii=False))
{"affected_count": 5}
```

参数名	参数含义
affected_count	返回更新的 Record 数量。

删除 Record

最近更新时间：2025-09-12 21:41:52

接口定义

本接口 `delete_record()` 用于删除指定的长期记忆。

```
def delete_record(
    session_id: str,
    record_id: str,
    memory_id: str | None = None,
    actor_id: str | None = None
) -> Dict[str, Any]
```

使用示例

```
import json
from tdaimemory import MemoryClient

del_record_result = client.delete_record(
    session_id="session-Y5oi****",
    record_id="record-RJTh****",
)

print(json.dumps(del_record_result, ensure_ascii=False))
```

请求参数

参数	是否必选	参数含义	配置方法及要求
memory_id	否	指定需删除 Record 的 Memory ID。	登录 Memory 智能体控制台 ，可在 Memory 服务概览页面复制 Memory ID。具体可以参见 查看 Memory 。
actor_id	否	指定需删除 Record 的用户 ID。	<ul style="list-style-type: none">数据类型：String长度限制：[1,20]字符规则：任意字符
session	是	指定需删除 Record 的会话	获取会话 ID，请参见 查询 Session 。

<code>_id</code>		ID。	
<code>record_id</code>	是	指定需删除的 Record ID。	获取 Record ID，请参见 查询 Record 。

响应消息

执行成功，输出如下信息。

```
>>> print(json.dumps(del_record_result, ensure_ascii=False))
{"affected_count": 2}
```

参数名	参数含义
<code>affected_count</code>	返回删除的 Record 数量。

搜索 Record

最近更新时间：2025-09-12 21:41:52

接口定义

本接口 `search_records()` 用于根据关键词语义与策略类型检索相关的长期记忆，并给出相关性得分。

```
def search_records(
    content: str,
    memory_id: str | None = None,
    actor_id: str | None = None,
    session_id: str | None = None,
    strategies: List[str] | None = None,
    where: Dict[str, Any] | None = None,
    order_by: Dict[str, int] | None = None,
    limit: int | None = None
) -> Dict[str, Any]
```

使用示例

```
import json
from tdaimemory import MemoryClient

# 17. Search records
search_records = client.search_records(
    content="美食维度",
    session_id="session-Y5oi****",
    strategies=["旅游关注维度"],
    # where={"category": "docs"},
    order_by={"created_at": -1},
    limit=10
)
print(json.dumps(search_records, ensure_ascii=False))
```

请求参数

参数	是否必选	参数含义	配置方法及要求
----	------	------	---------

memory_id	否	指定搜索 Record 所关联的 Memory ID。	登录 Memory 智能体控制台 ，可在 Memory 服务概览页面复制 Memory ID。具体可以参见 查看 Memory 。
actor_id	否	指定搜索 Record 用户的唯一标识 ID。	<ul style="list-style-type: none"> 数据类型：String 长度限制：[1,20] 字符规则：任意字符
session_id	是	指定搜索 Record 的 Session ID。	获取会话 ID，请参见 查询 Session 。
content	否	指定搜索的关键词，用于匹配记录内容。	字符串类型，例如：用户偏好。
strategy	否	指定 Record 策略类型。	<p>数据类型为 String 数组。默认支持如下策略，可自定义其他策略：</p> <ul style="list-style-type: none"> Persona_Profile：人物画像。 Task_Information：任务信息。 Factual_Experience：事实经验。
where	否	<p>where 指定查询条件，返回完全符合这些规则的事件。</p> <div style="border: 1px solid #00aaff; padding: 5px; margin-top: 10px;"> <p>说明： 当前仅支持使用 record_id、created_at、updated_at 字段设置查询条件。</p> </div>	<p>设置规则为：以键值对的形式，定义事件消息体中字段的匹配规则。</p> <ul style="list-style-type: none"> 支持使用的操作符。 <ul style="list-style-type: none"> 比较操作符：\$eq（等于，默认可省略）、\$ne（不等于）、\$gt（大于）、\$gte（大于等于）、\$lt（小于）、\$lte（小于等于）。 集合操作符：\$in（包含在数组中）、\$nin（不包含在数组中） 如下表示查找 status 字段等于 "ACTIVE" 且 score 字段大于等于80的记录。 <pre style="background-color: #2d3748; color: #e2e8f0; padding: 10px; margin-top: 10px;"> { "where": { "status": "ACTIVE", "score": { "\$gte": 80 } } }</pre>
order_by	否	指定返回的 Record 的排序方式。	支持如下两种方式。

			<ul style="list-style-type: none">• -1: 降序。默认按照 Record 的创建时间降序排列。• 1: 升序。
limit	否	指定返回的 Record 数量。	<ul style="list-style-type: none">• 数据类型: int, 正整数。• 取值范围: 1~100。• 默认值: 10。
offset	否	指定返回结果列表的起始偏移量。	<ul style="list-style-type: none">• 数据类型: int, 正整数。• 取值范围: [0, ∞]。• 默认值: 0, 表示从数据集的第一条记录开始返回。• 示例: offset=20表示跳过前20条记录, 从第21条开始返回。

响应消息

执行成功, 输出如下信息。

```
>>> print(json.dumps(search_records, ensure_ascii=False))
{
  "records": [
    {
      "created_at": 1757414225,
      "event_ids": "[]",
      "record_content": "美食维度为地道风味。",
      "record_id": "record-Un4a****",
      "score": 0.7984535,
      "strategy_name": "旅游关注维度",
      "updated_at": 1757414225
    },
    {
      "created_at": 1757412738,
      "event_ids": "[\"event-EuY_****\"]",
      "record_content": "在旅游方面, [++]对美食特别感兴趣",
      "record_id": "record-vaWf****",
      "score": 0.5307081,
      "strategy_name": "旅游关注维度",
      "updated_at": 1757412738
    },
    {
      "created_at": 1757412743,
```

```

    "event_ids": "[\"event-EuY_****\"]",
    "record_content": "在旅游方面,对花生严重过敏",
    "record_id": "record-CaH****",
    "score": 0.40438196,
    "strategy_name": "旅游关注维度",
    "updated_at": 1757412743
  },
  {
    "created_at": 1757412740,
    "event_ids": "[\"event-EuY_****\"]",
    "record_content": "在旅游方面, [++] 酒店要带健身房",
    "record_id": "record-cQyh****",
    "score": 0.3934074,
    "strategy_name": "旅游关注维度",
    "updated_at": 1757412740
  },
  {
    "created_at": 1757412746,
    "event_ids": "[\"event-EuY_****\"]",
    "record_content": "女朋友更喜欢现代艺术",
    "record_id": "record-Wet-****",
    "score": 0.38902017,
    "strategy_name": "旅游关注维度",
    "updated_at": 1757412746
  }
],
"total_count": 5
}
...

```

参数	子参数	参数含义
records	record_id	长期记忆 ID, 唯一标识该记录的标识符。
	record_content	记录内容, 描述 Record 记忆的实际信息。
	strategy_name	生成该记录的策略名称,
	created_at	Record 创建时间。

	updated_at	Record 最后一次更新时间。
	score	相关性得分。
	event_ids	事件 ID。
total_count	-	符合查询条件的总 Record 数量。

HTTP API

API 简介

最近更新时间：2025-09-12 22:11:01

Agent Memory API 是一套为 AI Agent 提供记忆能力的核心服务接口，支持存储、检索和管理历史交互信息。该 API 通过统一的 POST + JSON 接口设计，旨在提供一致的用户体验和易于集成的开发接口。

接口层级	接口名	接口含义	请求方式	URL 拼接地址
Session	/memory/v1/session/create	创建 Session	POST	<code>https://{服务访问地址}/memory/v1/session/create</code>
	/memory/v1/session/query	查询 Session	POST	<code>https://{服务访问地址}/memory/v1/session/query</code>
	/memory/v1/session/update	更新 Session	POST	<code>https://{服务访问地址}/memory/v1/session/update</code>
	/memory/v1/session/delete	删除 Session	POST	<code>https://{服务访问地址}/memory/v1/session/delete</code>
	/memory/v1/session/search	搜索 Session	POST	<code>https://{服务访问地址}/memory/v1/session/search</code>
Event	/memory/v1/event/append	追加 Event	POST	<code>https://{服务访问地址}/memory/v1/event/append</code>

	/memory/v1/event/query	查询 Event	P O S T	https://{服务访问地址}/memory/v1/event/query
	/memory/v1/event/delete	删除 Event	P O S T	https://{服务访问地址}/memory/v1/event/delete
State	/memory/v1/state/set	设置 State	P O S T	https://{服务访问地址}/memory/v1/state/set
	/memory/v1/state/get	获取 State	P O S T	https://{服务访问地址}/memory/v1/state/get
	/memory/v1/state/get_all	获取所有 State	P O S T	https://{服务访问地址}/memory/v1/state/get_all
	/memory/v1/state/delete	删除 State	P O S T	https://{服务访问地址}/memory/v1/state/delete
	/memory/v1/state/flush	清空 State	P O S T	https://{服务访问地址}/memory/v1/state/flush
Record	/memory/v1/record/append	创建 Record	P O S T	https://{服务访问地址}/memory/v1/record/append
	/memory/v1/record/query	查询 Record	P O S T	https://{服务访问地址}/memory/v1/record/query

/memory/v1/record/update	更新 Record	P O S T	https://{服务访问地址}/memory/v1/record/update
/memory/v1/record/delete	删除 Record	P O S T	https://{服务访问地址}/memory/v1/record/delete
/memory/v1/record/search	搜索 Record	P O S T	https://{服务访问地址}/memory/v1/record/search

Session 管理

创建 Session

最近更新时间：2025-09-17 11:13:52

接口介绍

本接口 (`/memory/v1/session/create`) 用于创建一个新会话，并将其关联到指定的记忆库。

Method 与 URL

POST `https://{服务访问地址}/memory/v1/session/create`

使用示例

```
curl -i -k -X POST \  
  -H 'Content-Type: application/json' \  
  -H 'Authorization: Bearer *****' \  
  -H "x-tdai-service-id: tdai-mem-8i8t*****" \  
  https://memory.tdai.tencentyun.com/memory/v1/session/create \  
  -d '{  
    "memory_id": "tdai-mem-e3f7*****",  
    "actor_id": "user-test-001",  
    "name": "Travel_Consultation"  
  }'
```

请求参数

参数	是否必选	参数含义	配置方法及要求
memory_id	是	指定 Session 需关联的记忆库 ID。	登录 Memory 智能体控制台 可复制 Memory ID。具体信息，参见 查看 Memory 详情 。
actor_id	是	指定 Session 的用户 ID。	<ul style="list-style-type: none">数据类型：String长度限制：[1,20]字符规则：任意字符

name	否	指定 Session 名称。	<ul style="list-style-type: none"> ● 数据类型: String ● 长度限制: [1,128] ● 字符规则: 任意字符 ● 若不指定, 默认以 Session ID 命名。
------	---	----------------	---

响应消息

执行成功, 输出如下信息。

```

{
  "code": 0,
  "message": "Operation success",
  "data": {
    "session_id": "session-PTVM****"
  }
}

```

参数名 (一级)	参数名 (二级)	参数含义
data	session_id	为所创建的 Session 自动分配的具有唯一标识的 ID。

查询 Session

最近更新时间：2025-09-12 21:41:52

接口介绍

本接口 (/memory/v1/session/query) 用于获取与特定 Memory 和用户关联的所有 Session 列表。

Method 与 URL

POST https://{服务访问地址}/memory/v1/session/query

使用示例

```
curl -i -k -X POST \  
  -H 'Content-Type: application/json' \  
  -H 'Authorization: Bearer *****' \  
  -H "x-tdai-service-id: tdai-mem-8i8t****" \  
  https://memory.tdai.tencentyun.com/memory/v1/session/query \  
  -d '{  
    "memory_id": "tdai-mem-8i8t****",  
    "actor_id": "user-test-001",  
    "order_by": {  
      "created_at": -1  
    },  
    "limit": 10,  
    "offset": 0  
  }'
```

请求参数

参数	是否必选	参数含义	配置方法及要求
memory_id	是	指定查询 Session 需关联的记忆库 ID。	登录 Memory 智能体控制台 可复制 Memory ID。具体信息，参见 查看 Memory 详情 。
actor_id	是	指定查询 Session 的用户 ID。	<ul style="list-style-type: none">数据类型：String长度限制：[1,20]字符规则：任意字符

order_by	否	指定返回的 Session 的排序方式。	支持如下两种方式。 <ul style="list-style-type: none"> • -1: 降序。默认按照 Session 的创建时间降序排列。 • 1: 升序。
limit	否	指定返回的会话数量。	<ul style="list-style-type: none"> • 数据类型: int, 正整数。 • 取值范围: 1~100。 • 默认值: 10。
offset	否	指定返回结果列表的起始偏移量。	<ul style="list-style-type: none"> • 数据类型: int, 正整数。 • 默认值: 0, 表示从数据集的第一条记录开始返回。 • 示例: offset=20表示跳过前20条记录, 从第21条开始返回。

响应消息

执行成功，输出如下信息。

```
{
  "code": 0,
  "message": "",
  "data": {
    "sessions": [
      {
        "created_at": 1757322888,
        "updated_at": 1757579197,
        "session_id": "session-PTVM****",
        "session_name": "Travel_Consultation"
      }
    ],
    "total_count": 1
  }
}
```

参数名 (一级)	参数名 (二级)	参数含义
data	seesions	列出满足查询条件的所有 Session。 <ul style="list-style-type: none"> • session_id: Session 的唯一标识 ID。 • session_name: Session 的会话名称。 • created_at: Session 的创建时间。

		<ul style="list-style-type: none"> • updated_at: Session 的最后一次更新时间。
total_count	-	满足查询条件的 Session 数量。

更新 Session

最近更新时间：2025-09-12 21:41:52

接口介绍

本接口（/memory/v1/session/update）用于更新一个已存在 Session 的属性信息，例如，更新名称。

Method 与 URL

POST https://{服务访问地址}/memory/v1/session/update

使用示例

```
curl -i -k -X POST \  
  -H 'Content-Type: application/json' \  
  -H 'Authorization: Bearer *****' \  
  -H "x-tdai-service-id: tdai-mem-8i8t****" \  
  https://memory.tdai.tencentyun.com/memory/v1/session/update \  
  -d '{  
    "memory_id": "tdai-mem-8i8t****",  
    "actor_id": "user-test-001",  
    "session_id": "session-PTVM****",  
    "name": "Travel-Consultation"  
  }'
```

请求参数

参数	是否必选	参数含义	配置方法及要求
memory_id	是	指定所更新 Session 所关联的 Memory ID。	登录 Memory 智能体控制台 可复制 Memory ID。具体信息，参见 查看 Memory 详情 。
actor_id	是	指定所更新 Session 的用户 ID。	<ul style="list-style-type: none">数据类型：String长度限制：[1,20]字符规则：任意字符
session_id	是	指定所需要更新的 Session ID。	-
name	是	指定更新后的 Session 名称。	确认 Name 的要求

响应消息

执行成功，输出如下信息。

```
{
  "code": 0,
  "message": "Session updated successfully",
  "data": {
    "affected_count": 1
  }
}
```

参数名（一级）	参数名（二级）	参数含义
data	affected_count	返回变更 Session 的数量。

删除 Session

最近更新时间：2025-09-12 21:41:52

接口介绍

本接口（/memory/v1/session/delete）用于永久删除一个指定的会话（Session）及其相关数据。

⚠ 注意：

通过本接口执行删除操作后，指定会话的所有相关信息（包括但不限于会话中的事件记录、状态数据及其他关联数据）将被彻底清除，一旦执行将无法撤销或恢复。

Method 与 URL

POST https://{服务访问地址}/memory/v1/session/delete

使用示例

```
curl -i -k -X POST \  
  -H 'Content-Type: application/json' \  
  -H 'Authorization: Bearer *****' \  
  -H "x-tdai-service-id: tdai-mem-8i8t****" \  
  https://memory.tdai.tencentyun.com/memory/v1/session/delete \  
  -d '{  
    "memory_id": "tdai-mem-8i8t****",  
    "actor_id": "user-test-001",  
    "session_id": "session-PTVM****"  
  }'
```

请求参数

参数	是否必选	参数含义	配置方法及要求
memory_id	是	指定所需删除 Session 所关联的 Memory ID。	登录 Memory 智能体控制台 可复制 Memory ID。具体信息，参见 查看 Memory 详情 。
actor_id	是	指定删除 Session 的用户 ID。	<ul style="list-style-type: none">数据类型：String长度限制：[1,20]字符规则：任意字符

session_id	是	指定所需删除的 Session ID。	-
------------	---	---------------------	---

响应消息

执行成功，输出如下信息。

```
{
  "code": 0,
  "message": "Session deleted successfully",
  "data": {
    "affected_count": 1
  }
}
```

参数名（一级）	参数名（二级）	参数含义
data	affected_count	返回删除 Session 的数量。

搜索 Session

最近更新时间：2025-09-12 21:41:52

接口介绍

本接口 (/memory/v1/session/search) 提供基于关键词规则匹配的会话搜索功能，可根据会话名称检索并返回符合条件的会话列表。

Method 与 URL

POST <https://{服务访问地址}/memory/v1/session/search>

使用示例

```
curl -i -k -X POST \  
  -H 'Content-Type: application/json' \  
  -H 'Authorization: Bearer *****' \  
  -H "x-tdai-service-id: tdai-mem-8i8t****" \  
  https://memory.tdai.tencentyun.com/memory/v1/session/search \  
  -d '{  
    "memory_id": "tdai-mem-8i8t****",  
    "actor_id": "user-test-001",  
    "content": "Travel",  
    "order_by": {  
      "created_at": -1  
    },  
    "limit": 3  
  }'
```

请求参数

参数	是否必选	参数含义	配置方法及要求
memory_id	是	指定查询 Session 需关联的记忆库 ID。	登录 Memory 智能体控制台 可复制 Memory ID。具体信息，参见 查看 Memory 详情 。
actor_id	是	指定查询 Session 的用户 ID。	<ul style="list-style-type: none">数据类型：String长度限制：[1,20]字符规则：任意字符
content	是	指定搜索的关键词。	数据类型：String。

			限制：当前仅支持使用 Session Name 中包含的关键词。
order_by	否	指定返回的 Session 的排序方式。	支持如下两种方式。 <ul style="list-style-type: none"> -1: 降序。默认按照 Session 的创建时间降序排列。 1: 升序。
limit	否	返回的最大会话数量。	<ul style="list-style-type: none"> 数据类型: int。 默认值: 3 取值范围: [1,50]。

响应消息

执行成功，输出如下信息。

```
{
  "code": 0,
  "message": "",
  "data": {
    "sessions": [
      {
        "created_at": 1757322888,
        "updated_at": 1757579197,
        "session_id": "session-PTVM****",
        "session_name": "Travel_Consultation"
      }
    ],
    "total_count": 1
  }
}
```

参数名（一级）	参数名（二级）	参数含义
data	seesions	列出满足查询条件的 Session。 <ul style="list-style-type: none"> session_id: Session 的唯一标识 ID。 session_name: Session 的会话名称。 created_at: Session 的创建时间。 updated_at: Session 的最后一次更新时间。
total_cou	-	满足查询条件的 Session 数量。

nt		
----	--	--

Event 管理

追加 Event

最近更新时间：2025-09-12 21:41:52

接口介绍

本接口（/memory/v1/event/append）用于向指定会话（Session）中添加新的对话事件。

说明：

Event 是用于记录一次完整的用户交互（如一轮对话或工具调用）的信息单元。

Method 与 URL

POST https://{服务访问地址}/memory/v1/event/append

使用示例

```
curl -i -k -X POST \  
-H 'Content-Type: application/json' \  
-H 'Authorization: Bearer *****' \  
-H "x-tdai-service-id: tdai-mem-8i8t****" \  
https://memory.tdai.tencentyun.com/memory/v1/event/append \  
-d '{  
  "memory_id": "tdai-mem-8i8t****",  
  "actor_id": "user-test-001",  
  "session_id": "session-PTVM****",  
  "messages": {  
    "chat": [  
      {  
        "role": "user",  
        "content": "嗨，帮我规划一个下周去杭州的3天行程。"  
      },  
      {  
        "role": "assistant",  
        "content": "好的，已为您推荐杭州三日行程。第一天，我们聚焦西湖经典。清晨从断桥残雪出发，漫步白堤，感受水墨画般的湖光山色。随后乘船登岛探访三潭印月，下午登雷峰塔俯瞰全景，傍晚在曲院风荷赏夕阳，晚上则去坊街夜市品尝地道小吃。\\n\\n第三天，探索湿地与运河文化。上午在西溪湿地乘坐摇橹船，听船娘小调，感受水乡慢生活；下午逛一逛桥西历史街区和文艺的小河直街，参观工艺美术博物馆，在咖啡馆小憩后即可轻松结束旅程，准备返程。\\n\\n第二天，体验禅意与宋韵。上午前往灵隐寺景区祈福，在法喜寺的黄墙拍照打卡；下午沉
```

浸于宋城，体验AR剧本杀并观看《宋城千古情》演出；如果傍晚还有精力，可以去九溪烟树享受清
凉的溪水。"

```
    },  
    {  
      "role": "user",  
      "content": "我喜欢人少、安静的地方，对美食特别感兴趣。"  
    },  
    {
```

```
      "role": "assistant",  
      "content": "基于您喜欢安静和美食的偏好，我为您优化了行程，避开了  
团队游客扎堆的景点，专注于幽静的体验和地道的风味。\\n\\n第一天：避世西湖\\n清晨漫步杨公堤  
与茅家埠，享受静谧湖景。午餐在老字号「奎元馆」吃虾爆鳝面。下午乘船访三潭印月，傍晚在湖  
边茶社小坐。晚餐去本地人爱的胜利河美食街尝地道风味。\\n\\n第二天：禅意之路\\n一早探灵隐  
寺，然后沿幽静天竺路步行至法喜寺。午餐在「知竹」吃素面。下午深入云栖竹径看竹海。晚餐用  
「新丰小吃」的喉口馒头等市井味道收尾。\\n\\n第三天：水乡慢生活\\n全天在西溪湿地坐摇橹船，  
深入芦苇荡享受宁静。午后往文艺的小河直街，在运河畔的咖啡馆小坐休憩，结束旅程。"
```

```
    },  
    {  
      "role": "user",  
      "content": "太好了，酒店要带健身房，另外我对花生严重过敏，预订餐  
厅时请务必注意。"
```

```
    },  
    {  
      "role": "assistant",  
      "content": "好的，已完全了解您的需求（偏好安静人少、喜爱美食、酒  
店需带健身房、对花生严重过敏）。为您精选的酒店（带健身房）：\\n\\n西湖庐驿精品酒店：位于  
西湖区，设计静谧，配备健身房，且远离闹市，符合您对安静的需求。\\n\\n杭州西溪紫金港希尔顿  
逸林酒店：靠近西溪湿地，设施完善，健身房专业，适合在第三天安排湿地行程时入住。"
```

```
    },  
    {  
      "role": "user",  
      "content": "完美！哦对了，我女朋友更喜欢现代艺术，有什么地方可以  
推荐吗？"
```

```
    },  
    {  
      "role": "assistant",  
      "content": "收到！考虑到您女朋友的喜好。这是为您和您女友定制的杭  
州三日行程：\\n\\n第一天：上午游茅家埠、杨公堤；下午观天目里美术馆现代展览。晚餐于胜利河  
美食街 品尝安全地道的本帮菜。\\n\\n第二天：上午早访灵隐寺，漫步幽静天竺路。下午重点参观  
浙江省博物馆之江馆 重磅国际艺术大展。晚餐体验本地 新丰小吃（已规避风险）。\\n\\n第三天：  
上午于 西溪湿地乘摇橹船。下午在小河直街 咖啡馆悠闲小憩，结束旅程。"
```

```
  ]
```

```

}
}'

```

请求参数

参数	是否必选	参数含义	配置方法及要求
memory_id	是	指定追加事件的会话 (Session) 所关联的 Memory ID。	登录 Memory 智能体控制台 可复制 Memory ID。具体信息，参见 查看 Memory 详情 。
actor_id	是	指定追加事件的会话 (Session) 中所属用户的唯一标识 ID。	<ul style="list-style-type: none"> 数据类型: String 长度限制: [1,20] 字符规则: 任意字符
session_id	是	指定追加 Event 的 Session ID。	获取 Session ID，请参见 查询 Session 。
messages	是	指定追加事件的消息对象。	<p>以 json 格式配置消息体。本示例以 role (角色) 和 content (内容) 两个字段，记录事件中不同参与者 (如用户、助手、工具、系统) 的具体交互内容。</p> <ul style="list-style-type: none"> role: 指定发言者身份。 <ul style="list-style-type: none"> user: 表示最终用户。 assistant: AI 助手。 tool: 被调用的工具或函数。 system: 系统指令或通知。 content: 对应角色发出的实际文本内容或数据。

响应消息

执行成功，输出如下信息。

```

{
  "code": 0,
  "message": "Event appended successfully",
  "data": {
    "event_id": "event-Ej9s****"
  }
}

```

参数名（一级）	参数名（二级）	参数含义
data	event_id	返回创建的事件 ID。

查询 Event

最近更新时间：2025-09-12 21:41:52

接口介绍

本接口（`/memory/v1/event/query`）用于查询会话中的事件。

Method 与 URL

POST `https://{服务访问地址}/memory/v1/event/query`

使用示例

```
curl -i -k -X POST \  
-H 'Content-Type: application/json' \  
-H 'Authorization: Bearer *****' \  
-H "x-tdai-service-id: tdai-mem-8i8t****" \  
https://memory.tdai.tencentyun.com/memory/v1/event/query \  
-d '{  
  "memory_id": "tdai-mem-8i8t****",  
  "actor_id": "user-test-001",  
  "session_id": "session-PTVM****",  
  "order_by": {  
    "created_at": -1  
  },  
  "limit": 5,  
  "offset": 0  
}'
```

请求参数

参数	是否必选	参数含义	配置方法及要求
memory_id	是	指定查询会话（Session）所关联的 Memory ID。	登录 Memory 智能体控制台 可复制 Memory ID。具体信息，参见 查看 Memory 详情 。
actor_id	是	指定查询会话（Session）中所属用户的唯一标识 ID。	<ul style="list-style-type: none">数据类型：String长度限制：[1,20]字符规则：任意字符

session_id	是	指定查询事件的 Session ID。	获取 Session ID，请参见 查询 Session 。
where	否	指定查询条件的对象，通过键值对指定字段的匹配规则，返回满足指定条件的事件。	<ul style="list-style-type: none"> 支持使用的操作符。 <ul style="list-style-type: none"> 比较操作符：\$eq（等于，默认可省略）、\$ne（不等于）、\$gt（大于）、\$gte（大于等于）、\$lt（小于）、\$lte（小于等于）。 集合操作符：\$in（包含在数组中）、\$nin（不包含在数组中） 查询条件设置示例。 如下表示查找 status 字段等于 "ACTIVE" 且 score 字段大于等于80的记录。 <pre> { "where": { "status": "ACTIVE", "score": { "\$gte": 80 } } }</pre>
order_by	否	指定返回的 Event 的排序方式。	支持如下两种方式。 <ul style="list-style-type: none"> -1: 降序。默认按照 Event 的创建时间降序排列。 1: 升序。
limit	否	指定返回的事件数量。	<ul style="list-style-type: none"> 数据类型：int，正整数。 取值范围：1~100。 默认值：10。
offset	否	指定返回结果列表的起始偏移量。	<ul style="list-style-type: none"> 数据类型：int，正整数。 默认值：0，表示从数据集的第一条记录开始返回。 示例：offset=20表示跳过前20条记录，从第21条开始返回。

响应消息

执行成功，输出如下信息。

```
{
  "code": 0,
  "message": "",
  "data": {
    "events": [{
      "chat": [{
        "content": "嗨，帮我规划一个下周去杭州的3天行程。",
        "role": "user"
      }],
      {
        "content": "好的，已为您推荐杭州三日行程。第一天，我们聚焦西湖经典。清晨从断桥残雪出发，漫步白堤，感受水墨画般的湖光山色。随后乘船登岛探访三潭印月，下午登雷峰塔俯瞰全景，傍晚在曲院风荷赏夕阳，晚上则去河坊街夜市品尝地道小吃。\\n\\n第三天，探索湿地与运河文化。上午在西溪湿地乘坐摇橹船，听船娘小调，感受水乡慢生活；下午逛一逛桥西历史街区 and 文艺的小河直街，参观工艺美术博物馆，在咖啡馆小憩后即可轻松结束旅程，准备返程。\\n\\n第二天。",
        "role": "assistant"
      }],
      {
        "content": "我喜欢人少、安静的地方，对美食特别感兴趣。",
        "role": "user"
      }],
      {
        "content": "基于您喜欢安静和美食的偏好，我为您优化了行程，避开了团队游客扎堆的景点，专注于幽静的体验和地道的风味。\\n\\n第一天：避世西湖 \\n清晨漫步杨公堤与茅家埠，享受静谧湖景。午餐在老字号「奎元馆」吃虾爆鳝面。下午乘船访三潭印月，傍晚在湖边茶社小坐。晚餐去本地人爱的胜利河美食街尝地道风味。\\n\\n第二天：禅意之路 \\n一早探灵隐寺，然后沿幽静天竺路步行至法喜寺。午餐在「知竹」吃素面。下午深入云栖竹径看竹海。晚餐用「新丰小吃」的喉口馒头等市井味'。",
        "role": "assistant"
      }],
      {
        "content": "太好了，酒店要带健身房，另外我对花生严重过敏，预订餐厅时请务必注意。",
        "role": "user"
      }],
      {
        "content": "好的，已完全了解您的需求（偏好安静人少、喜爱美食、酒店需带健身房、对花生严重过敏）。为您精选的酒店（带健身房）： \\n\\n西湖庐驿精品酒店：位于西湖区，设计静谧，配备健身房，且远离闹市，符合您对安静的需求。\\n\\n杭州西溪紫金港希尔顿逸林酒店：靠近西溪湿地，设施完善，健身房专业，适合在第三天行程时入住。",
        "role": "assistant"
      }],
      {

```

```

        "content": "完美！哦对了，我女朋友更喜欢现代艺术，有什么地方可以推荐吗？",
        "role": "user"
    },
    {
        "content": "收到！考虑到您女朋友的喜好。这是为您和您女友定制的杭州三日行程：\n\n第一天：上午游茅家埠、杨公堤；下午观天目里美术馆现代展览。晚餐于胜利河美食街品尝安全地道的本帮菜。\n\n第二天：上午早访灵隐寺，漫步幽静天竺路。下午重点参观 浙江省博物馆之江馆 重磅国际艺术大展。晚餐体验本地 新丰小吃（已规避风险）。\n\n第三天：上午于西溪湿地乘摇橹船。下午在小河直街 咖啡馆悠闲小憩，结束旅程。",
        "role": "assistant"
    }
  ],
  "created_at": 1757384362,
  "event_id": "event-Ej9s****",
  "updated_at": 1757384362
}],
"total_count": 1
}
}

```

参数名 (一级)	参数名 (二级)	参数名 (三级)	参数含义
data	events	event_id	返回满足条件的事件 ID。
		messages	返回事件的消息对象。
		created_at	事件创建时间。
	total_count	-	符合条件的事件总数。

删除 Event

最近更新时间：2025-09-12 21:41:52

接口介绍

本接口（ /memory/v1/event/delete ）用于从记忆库中永久删除指定的事件记录及相关数据。

⚠ 注意：

删除事件可能影响记忆库中的对话历史或短期记忆的内容。

Method 与 URL

POST https://{服务访问地址}/memory/v1/event/delete

使用示例

```
curl -i -k -X POST \  
  -H 'Content-Type: application/json' \  
  -H 'Authorization: Bearer *****' \  
  -H "x-tdai-service-id: tdai-mem-8i8t****" \  
  https://memory.tdai.tencentyun.com/memory/v1/event/delete \  
  -d '{  
    "memory_id": "tdai-mem-8i8t****",  
    "actor_id": "user-test-001",  
    "session_id": "session-PTVM****",  
    "event_id": "event-Ej9s****"  
  }'
```

请求参数

参数	是否必选	参数含义	配置方法及要求
memory_id	是	指定需删除事件所属会话所关联的 Memory ID。	登录 Memory 智能体控制台 可复制 Memory ID。具体信息，参见 查看 Memory 详情 。
actor_id	是	指定需删除事件所属会话的用户 ID。	<ul style="list-style-type: none">数据类型：String长度限制：[1,20]字符规则：任意字符

session_id	是	指定需删除事件所属会话 ID。	获取 Session ID，请参见 查询 Session 。
event_id	是	指定需删除的具体事件 ID。	获取 Event ID，请参见 查询 Event 。

响应消息

执行成功，输出如下信息。

```
{
  "code": 0,
  "message": "Event deleted successfully",
  "data": {
    "affected_count": 1
  }
}
```

参数名（一级）	参数名（二级）	参数含义
data	affected_count	返回删除 Event 的数量。

State 管理

设置 State

最近更新时间：2025-09-12 21:41:52

接口介绍

本接口 (/memory/v1/state/set) 用于设置当前会话中的状态数据。

Method 与 URL

```
POST https://{服务访问地址}/memory/v1/state/set
```

使用示例

```
curl -i -k -X POST \  
  -H 'Content-Type: application/json' \  
  -H 'Authorization: Bearer *****' \  
  -H "x-tdai-service-id: tdai-mem-8i8t****" \  
  https://memory.tdai.tencentyun.com/memory/v1/state/set \  
  -d '{  
    "memory_id": "tdai-mem-8i8t****",  
    "actor_id": "user-test-001",  
    "session_id": "session-PTVM****",  
    "state": {  
      "环境偏好": "安静、人少",  
      "天气查询": "查询完成",  
      "notification_enabled": true,  
      "user_language": "zh-CN"  
    }  
  }'
```

请求参数

参数	是否必选	参数含义	配置方法及要求
memory_id	是	指定需设置当前会话状态的 Memory ID。	登录 Memory 智能体控制台 可复制 Memory ID。具体信息，参见 查看 Memory 详情 。

actor_id	是	指定需设置当前会话状态的用户 ID。	<ul style="list-style-type: none"> 数据类型: String 长度限制: [1,20] 字符规则: 任意字符
session_id	是	指定需设置当前会话状态的会话 ID。	获取 Session ID, 请参见 查询 Session 。
state	是	指定需设置的状态数据, 对于已存在的更新, 不存在的新增。	<p>允许包含一个或多个任意类型的键值对, 如字符串、数字、布尔值、数组、对象等。示例如下所示。</p> <pre> { "user_preference_theme": "light", // 字符串值 "shopping_cart_items": ["laptop", "headphones"], // 数组/列表值 "user_language": "zh-CN", // 字符串值 "notification_enabled": true // 布尔值 } </pre>

响应消息

执行成功, 输出如下信息。

```

{
  "code": 0,
  "message": "State set successfully",
  "data": {
    "affected_count": 4
  }
}

```

参数名（一级）	参数名（二级）	参数含义
data	affected_count	设置成功的状态数量。

获取 State

最近更新时间：2025-09-12 21:41:52

接口介绍

本接口 (/memory/v1/state/get) 用于获取当前会话中的状态数据。

Method 与 URL

POST https://{服务访问地址}/memory/v1/state/get

使用示例

```
curl -i -k -X POST \  
  -H 'Content-Type: application/json' \  
  -H 'Authorization: Bearer *****' \  
  -H "x-tdai-service-id: tdai-mem-8i8t****" \  
  https://memory.tdai.tencentyun.com/memory/v1/state/get \  
  -d '{  
    "memory_id": "tdai-mem-8i8t****",  
    "actor_id": "user-test-001",  
    "session_id": "session-PTVM****",  
    "keys": ["user_language", "环境偏好"]  
  }'
```

请求参数

参数	是否必选	参数含义	配置方法及要求
memory_id	是	指定需获取当前会话状态的 Memory ID。	登录 Memory 智能体控制台 可复制 Memory ID。具体信息，参见 查看 Memory 详情 。
actor_id	是	指定需获取当前会话状态的用户 ID。	<ul style="list-style-type: none">数据类型：String长度限制：[1,20]字符规则：任意字符
session_id	是	指定需获取当前会话状态的会话 ID。	获取 Session ID，请参见 查询 Session 。
keys	是	指定需获取当前会话的状态键列表。	<ul style="list-style-type: none">数据类型：String 数组

- 示例: ["user_preference_theme", "shopping_cart_items"]

响应消息

执行成功，输出如下信息。

```
{
  "status": "success",
  "message": "State get successfully",
  "data": {
    "state": {
      "环境偏好": "安静、人少",
      "user_language": "zh-CN"
    }
  }
}
```

参数名（一级）	参数名（二级）	参数含义
data	state	返回状态数据 Key 对应的 Value。

获取所有 State

最近更新时间：2025-09-12 21:41:52

接口介绍

本接口 (/memory/v1/state/get_all) 用于获取当前会话中所有状态数据。

Method 与 URL

POST https://{服务访问地址}/memory/v1/state/get_all

使用示例

```
curl -i -k -X POST \  
-H 'Content-Type: application/json' \  
-H 'Authorization: Bearer *****' \  
-H "x-tdai-service-id: tdai-mem-8i8t****" \  
https://memory.tdai.tencentyun.com/memory/v1/state/get_all \  
-d '{  
  "memory_id": "tdai-mem-8i8t****",  
  "actor_id": "user-test-001",  
  "session_id": "session-PTVM****"  
}'
```

请求参数

参数	是否必选	参数含义	配置方法及要求
memory_id	是	指定需获取当前会话状态的 Memory ID。	登录 Memory 智能体控制台 可复制 Memory ID。具体信息，参见 查看 Memory 详情 。
actor_id	是	指定需获取当前会话状态的用户 ID。	<ul style="list-style-type: none">数据类型：String长度限制：[1,20]字符规则：任意字符
session_id	是	指定需获取当前会话状态的会话 ID。	获取 Session ID，请参见 查询 Session 。

响应消息

执行成功，输出如下信息。

```

{
  "code": "0",
  "message": "All states get successfully",
  "data": {
    "state": {
      "user_language": "zh-CN",
      "notification_enabled": true,
      "环境偏好": "安静、人少",
      "天气查询": "查询完成"
    }
  }
}
    
```

参数名（一级）	参数名（二级）	参数含义
data	state	返回会话中所有状态键值。

删除 State

最近更新时间：2025-09-12 21:41:52

接口介绍

本接口（/memory/v1/state/delete）用于删除会话中指定的一个或多个状态键。

说明：

本接口仅删除指定的状态键，不会清空整个会话中的所有状态信息。

Method 与 URL

POST https://{服务访问地址}/memory/v1/state/delete

使用示例

```
curl -i -k -X POST \  
  -H 'Content-Type: application/json' \  
  -H 'Authorization: Bearer *****' \  
  -H "x-tdai-service-id: tdai-mem-8i8t****" \  
  https://memory.tdai.tencentyun.com/memory/v1/state/delete \  
  -d '{  
    "memory_id": "tdai-mem-8i8t****",  
    "actor_id": "user-test-001",  
    "session_id": "session-PTVM****",  
    "keys": ["notification_enabled", "天气查询"]  
  }'
```

请求参数

参数	是否必选	参数含义	配置方法及要求
memory_id	是	指定需删除会话状态的 Memory ID。	登录 Memory 智能体控制台 可复制 Memory ID。具体信息，参见 查看 Memory 详情 。
actor_id	是	指定需删除会话状态的用户 ID。	<ul style="list-style-type: none">数据类型：String长度限制：[1,20]字符规则：任意字符

session_id	是	指定需删除会话状态的会话 ID。	获取 Session ID，请参见 查询 Session 。
keys	是	指定需删除的状态键列表。	<ul style="list-style-type: none"> 数据类型：String 数组。 示例：["user_preference_theme", "shopping_cart_items"]

响应消息

执行成功，输出如下信息。

```

{
  "code": "0",
  "message": "All states deleted successfully",
  "data": {
    "affected_count": 2
  }
}

```

参数名（一级）	参数名（二级）	参数含义
data	affected_count	返回删除的状态键数量。

清空 State

最近更新时间：2025-09-12 21:41:52

接口介绍

本接口 (/memory/v1/state/flush) 用于清空会话中的所有状态数据。

⚠ 注意:

本接口功能为删除当前会话中的全部状态键值对。此操作执行后数据将不可恢复，请您务必谨慎确认！

Method 与 URL

```
POST https://{服务访问地址}/memory/v1/state/flush
```

使用示例

```
curl -i -k -X POST \  
-H 'Content-Type: application/json' \  
-H 'Authorization: Bearer *****' \  
-H "x-tdai-service-id: tdai-mem-8i8t****" \  
https://memory.tdai.tencentyun.com/memory/v1/state/flush \  
-d '{  
  "memory_id": "tdai-mem-8i8t****",  
  "actor_id": "user-001",  
  "session_id": "session-PTVM****"  
}'
```

请求参数

参数	是否必选	参数含义	配置方法及要求
memory_id	是	指定需清空会话状态的 Memory ID。	登录 Memory 智能体控制台 可复制 Memory ID。具体信息，参见 查看 Memory 详情 。
actor_id	是	指定需清空会话状态的用户 ID。	<ul style="list-style-type: none">数据类型：String长度限制：[1,20]字符规则：任意字符

session_id	是	指定需清空会话状态的会话 ID。	获取 Session ID，请参见 查询 Session 。
------------	---	------------------	--

响应消息

执行成功，输出如下信息。

```
{
  "code": "0",
  "message": "All states deleted successfully",
  "data": {
    "affected_count": 2
  }
}
```

参数名（一级）	参数名（二级）	参数含义
data	affected_count	返回清空的状态键数量。

Record 管理

创建 Record

最近更新时间：2025-09-17 15:27:42

接口介绍

本接口 (/memory/v1/record/append) 用于创建新的长期记忆记录，用于存储用户自定义的个性化信息，如用户偏好、重要事实等。

Method 与 URL

```
POST https://{服务访问地址}/memory/v1/record/append
```

使用示例

```
curl -i -k -X POST \  
  -H 'Content-Type: application/json' \  
  -H 'Authorization: Bearer *****' \  
  -H "x-tdai-service-id: tdai-mem-8i8t****" \  
  https://memory.tdai.tencentyun.com/memory/v1/record/append \  
  -d '{  
    "memory_id": "tdai-mem-8i8t****",  
    "actor_id": "user-test-001",  
    "session_id": "session-PTVM****",  
    "strategy": "旅游关注维度",  
    "content": "美食维度为地道风味。"  
  }'
```

请求参数

参数	是否必选	参数含义	配置方法及要求
memory_id	是	指定需创建长期记忆的 Memory ID。	登录 Memory 智能体控制台 可复制 Memory ID。具体信息，参见 查看 Memory 详情 。
actor_id	是	指定需创建长期记忆的用户 ID。	<ul style="list-style-type: none">数据类型：String长度限制：[1,20]字符规则：任意字符

session_id	是	指定需创建长期记忆的会话 ID。	获取 Session ID，请参见 查询 Session 。
content	是	记录内容，描述记忆的实际信息。	<ul style="list-style-type: none"> 数据类型：字符串 示例："用户偏好使用深色主题，喜欢简洁的界面设计"。
strategy	是	指定长期记忆的策略类型。	<p>可设置的策略类型，默认支持如下策略，支持自定义其他策略：</p> <ul style="list-style-type: none"> Persona_Profile：人物画像。 Task_Information：任务信息。 Factual_Experience：事实经验。

响应消息

执行成功，输出如下信息。

```

{
  "status": "success",
  "message": "Record appended successfully",
  "data": {
    "record_id": "record-Ukuw****"
  }
}

```

参数名（一级）	参数名（二级）	参数含义
data	record_id	系统随机分配的长期记忆 ID，用于唯一标识该记录。

查询 Record

最近更新时间：2025-09-12 21:41:52

接口介绍

本接口（/memory/v1/record/query）用于查询指定条件下的长期记忆。

- 支持根据 Memory ID、Actor ID、Seesion ID等条件筛选 Record。
- 支持分页，便于管理和浏览记录数据。

Method 与 URL

POST https://{服务访问地址}/memory/v1/record/query

使用示例

```
curl -i -k -X POST \  
  -H 'Content-Type: application/json' \  
  -H 'Authorization: Bearer *****' \  
  -H "x-tdai-service-id: tdai-mem-8i8t****" \  
  https://memory.tdai.tencentyun.com/memory/v1/record/query \  
  -d '{  
    "memory_id": "tdai-mem-8i8t****",  
    "actor_id": "user-test-001",  
    "session_id": "session-PTVM****",  
    "order_by": {  
      "created_at": -1  
    },  
    "limit": 3,  
    "offset": 0  
  }'
```

请求参数

参数	是否必选	参数含义	配置方法及要求
memory_id	是	指定查询 Record 所关联的 Memory ID。	登录 Memory 智能体控制台 可复制 Memory ID。具体信息，参见 查看 Memory 详情 。

actor_id	是	指定查询 Record 用户的唯一标识 ID。	<ul style="list-style-type: none"> 数据类型: String 长度限制: [1,20] 字符规则: 任意字符
session_id	是	指定查询 Record 的 Session ID。	获取 Session ID, 请参见 查询 Session 。
where	是	<p>指定查询条件的对象, 通过键值对指定字段的匹配规则, 返回满足指定条件的事件。</p> <div style="border: 1px solid #00aaff; padding: 5px; margin-top: 10px;"> <p>说明: 当前仅支持使用 record_id、created_at、updated_at 字段设置查询条件。</p> </div>	<ul style="list-style-type: none"> 支持使用的操作符。 <ul style="list-style-type: none"> 比较操作符: \$eq (等于, 默认可省略)、\$ne (不等于)、\$gt (大于)、\$gte (大于等于)、\$lt (小于)、\$lte (小于等于)。 集合操作符: \$in (包含在数组中)、\$nin (不包含在数组中) 查询条件设置示例。 <pre style="background-color: #2d3748; color: #a6c9ec; padding: 10px; margin-top: 10px;">{ "where": { "created_at": {"\$gte": 1757398345}, "record_id": "record-Ukuw****" } }</pre>
order_by	否	指定返回的 Record 的排序方式。	<p>支持如下两种方式。</p> <ul style="list-style-type: none"> -1: 降序。默认按照 Record 的创建时间降序排列。 1: 升序。
limit	否	指定返回的 Record 数量。	<ul style="list-style-type: none"> 数据类型: int, 正整数。 取值范围: 1~100。 默认值: 10。
offset	否	指定返回结果列表的起始偏移量。	<ul style="list-style-type: none"> 数据类型: int, 正整数。 默认值: 0, 表示从数据集的第一条记录开始返回。 示例: offset=20表示跳过前20条记录, 从第21条开始返回。

响应消息

执行成功，输出如下信息。

```
{
  "code": 0,
  "message": "",
  "data": {
    "records": [
      {
        "created_at": 1757384673,
        "event_ids": ["event-Ej9s****"],
        "record_content": "用户对人少、安静的地方有强烈的偏好",
        "record_id": "record-rCYY****",
        "strategy_name": "User Preference Strategy",
        "updated_at": 1757384673
      },
      {
        "created_at": 1757384676,
        "event_ids": ["event-Ej9s****"],
        "record_content": "用户对美食有浓厚的兴趣",
        "record_id": "record-ChVi****",
        "strategy_name": "User Preference Strategy",
        "updated_at": 1757384676
      },
      {
        "created_at": 1757384678,
        "event_ids": ["event-Ej9s****"],
        "record_content": "用户对花生严重过敏，需要特别注意",
        "record_id": "record-4gbj****",
        "strategy_name": "User Preference Strategy",
        "updated_at": 1757384678
      }
    ],
    "total_count": 18
  }
}
```

参数名	子参数（一级）	子参数（二级）	参数含义
data	records	record_id	长期记忆 ID，唯一标识该记录的标识符。
		record_content	记录内容，描述 Record 记忆的实际信息。

	strategy_name	生成该记录的策略名称。
	created_at	Record 创建时间。
	updated_at	Record 最后一次更新时间。
total_count	-	符合查询条件的总 Record 数量。

更新 Record

最近更新时间：2025-09-12 21:41:52

接口介绍

本接口（ /memory/v1/record/update ）用于更新已存在的 Record 的信息。

Method 与 URL

POST https://{服务访问地址}/memory/v1/record/update

使用示例

```
curl -i -k -X POST \  
-H 'Content-Type: application/json' \  
-H 'Authorization: Bearer *****' \  
-H "x-tdai-service-id: tdai-mem-8i8t****" \  
https://memory.tdai.tencentyun.com/memory/v1/record/update \  
-d '{  
  "memory_id": "tdai-mem-8i8t****",  
  "actor_id": "user-test-001",  
  "session_id": "session-PTVM****",  
  "record_id": "record-81SQ****",  
  "content": "美食维度为地道风味优先, 排除花生过敏风险."  
}'
```

请求参数

参数	是否必选	参数含义	配置方法及要求
memory_id	是	指定需更新 Record 的 Memory ID。	登录 Memory 智能体控制台 可复制 Memory ID。具体信息，参见 查看 Memory 详情 。
actor_id	是	指定需更新 Record 的用户 ID。	<ul style="list-style-type: none">数据类型：String长度限制：[1,20]字符规则：任意字符
session_id	是	指定需更新 Record 的会话 ID。	获取 Session ID，请参见 查询 Session 。

record_id	是	指定需更新 Record ID。	获取 Record ID，请参见 查询 Record 。
content	是	编辑更新后的记录内容。	<ul style="list-style-type: none">数据类型：String。支持输入中文、英文等任意字符。

响应消息

执行成功，输出如下信息。

```
{
  "status": "success",
  "message": "Record updated successfully",
  "data": {
    "affected_count": 6
  }
}
```

参数名	参数名	参数含义
data	affected_count	返回更新的 Record 数量。

删除 Record

最近更新时间：2025-09-12 21:41:52

接口介绍

本接口（/memory/v1/record/delete）用于删除指定的长期记忆。

⚠ 注意：

本接口删除长期记忆，一旦操作，不可恢复，该记录将无法再次访问，请您务必谨慎操作！

Method 与 URL

POST https://{服务访问地址}/memory/v1/record/delete

使用示例

```
curl -i -k -X POST \  
  -H 'Content-Type: application/json' \  
  -H 'Authorization: Bearer *****' \  
  -H "x-tdai-service-id: tdai-mem-8i8t****" \  
  https://memory.tdai.tencentyun.com/memory/v1/record/delete \  
  -d '{  
    "memory_id": "tdai-mem-8i8t****",  
    "actor_id": "user-test-001",  
    "session_id": "session-PTVM****",  
    "record_id": "record-81SQ****"  
  }'
```

请求参数

参数	是否必选	参数含义	配置方法及要求
memory_id	是	指定需删除 Record 的 Memory ID。	登录 Memory 智能体控制台 可复制 Memory ID。具体信息，参见 查看 Memory 详情 。
actor_id	是	指定需删除 Record 的用户 ID。	<ul style="list-style-type: none">数据类型：String长度限制：[1,20]字符规则：任意字符

session_id	是	指定需删除 Record 的会话 ID。	获取 Session ID，请参见 查询 Session 。
record_id	是	指定需删除的 Record ID。	获取 Record ID，请参见 查询 Record 。

响应消息

执行成功，输出如下信息。

```
{
  "code": "0",
  "message": "Record deleted successfully"
  "data": {
    "affected_count": 1
  }
}
```

参数名	参数名	参数含义
data	affected_count	返回删除的 Record 数量。

搜索 Record

最近更新时间：2025-09-12 21:41:52

接口介绍

本接口（/memory/v1/record/search）用于根据关键词语义与策略类型检索相关的长期记忆，并给出相关性得分。

Method 与 URL

POST https://{服务访问地址}/memory/v1/record/search

使用示例

```
curl -i -k -X POST \  
  -H 'Content-Type: application/json' \  
  -H 'Authorization: Bearer *****' \  
  -H "x-tdai-service-id: tdai-mem-8i8t****" \  
  https://memory.tdai.tencentyun.com/memory/v1/record/search \  
  -d '{  
    "memory_id": "tdai-mem-8i8t****",  
    "actor_id": "user-test-001",  
    "session_id": "session-PTVM****",  
    "strategy": ["旅游关注维度"],  
    "content": "美食维度",  
    "order_by": {  
      "created_at": -1  
    },  
    "limit": 5,  
    "offset": 0  
  }'
```

请求参数

参数	是否必选	参数含义	配置方法及要求
memory_id	是	指定搜索 Record 所关联的 Memory ID。	登录 Memory 智能体控制台 可复制 Memory ID。具体信息，参见 查看 Memory 详情 。

actor_id	是	指定搜索 Record 用户的唯一标识 ID。	<ul style="list-style-type: none"> 数据类型: String 长度限制: [1,20] 字符规则: 任意字符
session_id	是	指定搜索 Record 的 Session ID。	获取 Session ID, 请参见 查询 Session 。
content	是	指定搜索的关键词, 用于匹配记录内容。	字符串类型, 例如: 用户偏好。
strategy	是	指定 Record 策略类型。	数据类型为 String 数组。默认支持如下策略, 可自定义其他策略: <ul style="list-style-type: none"> Semantic Strategy: 语义信息。 Summary Strategy: 事实总结。 User Preference Strategy: 用户偏好。
where	是	指定查询条件的对象, 通过键值对指定字段的匹配规则, 返回满足指定条件的事件。 <div style="border: 1px solid #00aaff; padding: 5px; margin-top: 10px;"> <p>说明: 当前仅支持使用 record_id、created_at、updated_at 字段设置查询条件。</p> </div>	<ul style="list-style-type: none"> 支持使用的操作符。 <ul style="list-style-type: none"> 比较操作符: \$eq (等于, 默认可省略)、\$ne (不等于)、\$gt (大于)、\$gte (大于等于)、\$lt (小于)、\$lte (小于等于)。 集合操作符: \$in (包含在数组中)、\$nin (不包含在数组中) 查询条件设置示例。 <pre style="background-color: #2d3748; color: #e2e8f0; padding: 10px; border-radius: 5px;">{ "where": { "created_at": {"\$gte": 1757398345}, "record_id": "record-Ukuw****" } }</pre>
order_by	否	指定返回的 Record 的排序方式。	支持如下两种方式。 <ul style="list-style-type: none"> -1: 降序。默认按照 Record 的创建时间降序排列。 1: 升序。
limit	否	指定返回的 Record 数量。	<ul style="list-style-type: none"> 数据类型: int, 正整数。

			<ul style="list-style-type: none"> 取值范围：1~100。 默认值：10。
offset	否	指定返回结果列表的起始偏移量。	<ul style="list-style-type: none"> 数据类型：int，正整数。 取值范围：[0, ∞]。 默认值：0，表示从数据集的第一条记录开始返回。 示例：offset=20表示跳过前20条记录，从第21条开始返回。

响应消息

执行成功，输出如下信息。

```

{
  "code": 0,
  "message": "",
  "data": {
    "records": [
      {
        "created_at": 1757384676,
        "event_ids": ["event-Ej9s****"],
        "record_content": "用户对美食有浓厚的兴趣",
        "record_id": "record-ChV****D",
        "score": 0.65423244,
        "strategy_name": "User Perference Strategy",
        "updated_at": 1757384676
      },
      {
        "created_at": 1757384695,
        "event_ids": ["event-Ej9s****"],
        "record_content": "用户喜欢人少、安静的地方，对美食特别感兴趣。",
        "record_id": "record-uJ4t****",
        "score": 0.5492037,
        "strategy_name": "Semantic Strategy",
        "updated_at": 1757384695
      },
      {
        "created_at": 1757384730,
        "event_ids": ["event-Ej9s****"],
        "record_content": "在2023年的杭州，在旅游方面，[++]对美食特别感兴趣。",
        "record_id": "record-Drsz****",
      }
    ]
  }
}

```

```

    "score": 0.5148556,
    "strategy_name": "旅游关注维度",
    "updated_at": 1757384730
  },
  {
    "created_at": 1757384698,
    "event_ids": ["event-Ej9s****"],
    "record_content": "用户对花生严重过敏，预订餐厅时需特别注意。",
    "record_id": "record-Z3ro****",
    "score": 0.47972357,
    "strategy_name": "Semantic Strategy",
    "updated_at": 1757384698
  },
  {
    "created_at": 1757384693,
    "event_ids": ["event-Ej9s****"],
    "record_content": "用户计划下周去杭州进行为期3天的旅行。",
    "record_id": "record-RCMU****",
    "score": 0.45423132,
    "strategy_name": "Semantic Strategy",
    "updated_at": 1757384693
  }
],
"total_count": 5
}

```

参数名	子参数（一级）	子参数（二级）	参数含义
data	records	record_id	长期记忆 ID，唯一标识该记录的标识符。
		record_content	记录内容，描述 Record 记忆的实际信息。
		strategy_name	生成该记录的策略名称，例如：User_Preference_Strategy、Semantic_Strategy 等。
		created_at	Record 创建时间。
		updated_at	Record 最后一次更新时间。

		score	相关性得分。
	total_count	-	符合查询条件的总 Record 数量。