CHAPTER

第1章

AI Agent 开发初体验

本章作为全书的开篇，将带领读者走进 AI Agent（智能体）的世界。首先从核心定义入手，厘清其本质与价值，包括 AI Agent 的定义、核心特征、与传统工具的区别及应用场景；然后介绍 AI Agent 架构的组成，并尝试使用扣子开发一个简单的智能体。

1.1 AI Agent 的定义与价值

本节主要介绍 AI Agent 的定义、核心价值及应用场景等内容。

1.1.1 什么是 AI Agent——OpenAI 的权威定义与核心特征

AI Agent 作为大模型能力落地的核心载体，逐渐成为行业的核心方向。其中，字节跳动独立研发的扣子（Coze）平台，具备低代码搭建、生态联动性强、落地效率高的优势，成为当下流行的 AI Agent 开发平台。在这样的行业背景下，我们有必要先厘清 AI Agent 的核心定义。

那么，什么是 AI Agent？这里，我们可以借用 OpenAI 对 AI Agent 的定义：AI Agent 是能够自主感知环境、规划任务、执行操作并根据反馈优化行为，以实现特定目标的智能体。这一定义打破了人们对传统 AI 工具的认知边界，凸显了“自主性”与“目标导向性”两大核心内核。

从上述定义出发，AI Agent 具备四大核心特征，如图 1-1 所示。这四大特征是 AI Agent 区别于普通 AI 应用的关键。

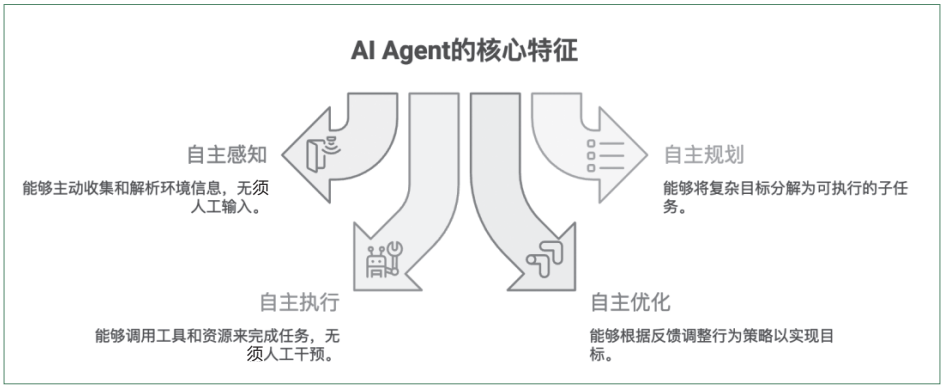


图 1-1 AI Agent 的核心特征

下面我们对 AI Agent 的核心特征进行详细的解释。

- 自主感知。AI Agent 能够主动收集并解析环境信息，这里的“环境”既可以是数字环境（如文档、网页、软件系统），也可以是物理环境（通过传感器获取现实场景数据）。与传统 AI 需要人工输入数据不同，AI Agent 能够主动识别和收集环境中的关键信息，为后续行动奠定基础。例如，办公场景中的 AI Agent 可以自动抓取邮件中的任务需求，无须人工筛选；智能家居场景中的 AI Agent 能够感知室内温湿度、光照等数据，进而识别出居住者的潜在需求。
- 自主规划。面对复杂目标，AI Agent 能够将其拆解为可执行的子任务，并制定合理的执行顺序。当目标较为模糊时，它还能通过追问等方式明确需求边界。比如，对于用户提出的“完成季度销售报告”，AI Agent 可拆解为“收集各区域销售数据”“整理数据并生成图表”“撰写报告正文”“校对优化格式”等子任务，并能根据数据获取的难易程度调整执行顺序。
- 自主执行。基于规划的任务清单，AI Agent 能够调用相应工具或资源完成具体操作，无须人工逐一干预。例如，在上述销售报告任务中，AI Agent 可自动连接企业 CRM（Customer Relationship Management，客户关系管理）系统提取数据，并使用表格软件生成图表，再通过文字生成工具撰写报告内容。这种执行能力不仅能完成单一任务，还能实现多任务的协同推进。
- 自主优化。AI Agent 能够根据执行结果与目标的偏差，结合环境反馈不断调整行为策略。当销售报告中的某组数据存在误差时，它能自动回溯数据来源，重新提取并修正；当用户对报告风格提出修改意见时，它可以快速调整撰写逻辑，优化输出结果。这种闭环优化能力让 AI Agent 能够适应动态变化的需求与环境。

1.1.2 AI Agent 与传统工具的本质区别——从“规则驱动”到“目标驱动”的跨越

要真正理解 AI Agent 的价值，必须厘清其与传统工具（包括传统 AI 工具）的本质区别。从核心逻辑来看，这一区别体现为从“规则驱动”到“目标驱动”的根本性跨越，如图 1-2 所示。

图 1-2 AI Agent 与传统工具的区别

传统工具，无论是基础的办公软件（如 Word、Excel）还是早期的 AI 应用（如简单的数据分析工具、语音识别软件），都遵循“规则驱动”的逻辑。这类工具的核心是“被动响应”——用户必须明确告知工具“做什么”以及“怎么做”，才能让工具按照预设的规则完成操作。以 Excel 为例，用户需要手动输入数据、设定公式、选择图表类型，每一个步骤都需要人工主导；再如早期的语音转文字工具，只能按照预设的语音识别规则将语音转换为文字，无法理解文字背后的语义，更无法根据用户需求对转换结果进行后续处理。

AI Agent 则完全不同，它以“目标驱动”为核心逻辑，实现了“主动服务”的突破。用户只需告知 AI Agent“要达成什么目标”，无须明确具体步骤，AI Agent 就能自主规划并完成任务。这种逻辑上的差异带来了两大关键变化：其一，责任主体的转移——传统工具的操作责任在用户，用户需要掌握工具的使用方法并确保步骤正确，而 AI Agent 的操作责任在自身，用户只需对最终目标负责；其二，能力边界的拓展——传统工具的能力局限于预设规则，无法处理规则之外的场景，而 AI Agent 通过自主感知、规划与优化，能够应对复杂多变的场景，甚至突破预设规则的限制，找到更优的任务解决方案。

举个具体的例子：针对“整理会议记录”这一需求，在使用传统语音转文字工具时，用户需要先将会议语音转换为文字，再手动筛选关键信息、梳理逻辑结构、划分任务分工；而在使用 AI Agent 时，用户只需上传会议语音并提出“整理会议记录，提取关键决议与任务分工”的目标，就能让 AI Agent 自动完成语音转文字、关键信息提取、逻辑梳理、格式优化等

一系列操作，最终输出结构化的会议记录。在这一过程中，用户从“工具操作者”转变为“目标设定者”，大幅降低了使用成本，提升了效率。

1.1.3 AI Agent 的核心价值——解放重复劳动、提升决策效率、拓展能力边界

基于“目标驱动”的核心逻辑与四大核心特征，AI Agent 在生产和生活中展现出三大核心价值，正在重塑生产力形态，推动效率革命。

第一大核心价值是解放重复劳动。在工作与生活中，存在大量机械、重复且低价值的劳动，如数据录入、文件整理、信息筛选、常规沟通等。这些劳动占用了人们大量的时间与精力，却难以创造核心价值。AI Agent 能够主动承接这类重复劳动，通过自主规划与执行，实现“自动化处理”。例如，对于企业行政人员的日常文件归档工作，AI Agent 可自动识别文件类型、提取关键信息、按照预设分类规则完成归档；在客服场景中，AI Agent 可自动回复常规咨询问题，处理订单查询、售后申请等基础业务，让客服人员专注于复杂问题的解决。这种对重复劳动的解放，让人们能够将精力投入创意、决策等更高价值的活动中。

第二大核心价值是提升决策效率。在复杂场景中，决策的难点往往在于信息繁杂、变量过多，难以快速精准地提炼关键信息并制定合理方案。AI Agent 具备强大的信息整合与分析能力，能够自主收集多源信息、梳理信息间的关联，并结合历史数据与场景需求生成多套决策方案和建议。例如，在企业市场推广决策中，AI Agent 可自动收集行业趋势、竞争对手动态、目标用户偏好等信息，分析不同推广渠道的效果数据，为企业制定精准的推广方案；在个人理财场景中，AI Agent 可整合个人收入、支出、资产状况等信息，结合市场理财产品动态，生成个性化的理财建议。通过 AI Agent 的辅助，决策过程从“信息筛选—分析研判—方案制定”的漫长流程，转变为“目标设定—方案评估—快速决策”的高效流程，大幅提升了决策的速度与准确性。

第三大核心价值是拓展能力边界。每个人的知识储备与技能水平都存在局限，而 AI Agent 能够将海量知识与多元技能整合为“个人能力延伸”，帮助人们完成原本难以完成的任务。例如，当普通职场人需要撰写英文商务邮件时，可使用 AI Agent 自动完成精准翻译与专业话术优化；当设计师需要了解某一新兴设计风格的核心要素时，可使用 AI Agent 快速整合相关资料并提炼关键要点；当学生需要梳理某一复杂学科的知识框架时，可使用 AI Agent 自动构建结构化知识体系。这种能力边界的拓展，不仅降低了专业门槛，更让人们能够快速适应多元需求，实现能力的“快速迭代”。

1.1.4 典型应用场景速览——工作、生活、娱乐中的 AI Agent 实例解析

AI Agent 的核心价值已在工作、生活、娱乐等多个场景中得到初步体现，不同场景的应用实例展现了其“目标驱动”逻辑的普适性，如图 1-3 所示。

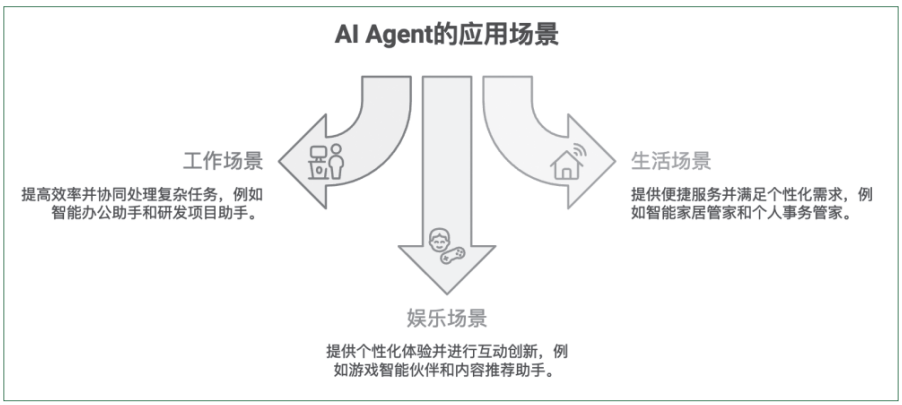


图 1-3 AI Agent 的应用场景

下面通过具体实例让读者快速感知 AI Agent 的应用价值。

在工作场景中，AI Agent 的应用聚焦于“效率提升”与“复杂任务协同”。智能办公助手就是典型的工作场景 AI Agent：只需用户设定“完成本周工作周报”的目标，它就可自动整合本周的工作邮件、会议记录、任务完成情况等信息，按照企业规定的周报格式，生成初稿后再根据用户的修改意见进行优化和完善；再如，研发项目助手能够自主跟踪项目进度，收集各研发环节的问题反馈，生成项目进度报告并提出风险预警，帮助项目负责人快速掌握项目动态。此外，在市场调研、客户管理、财务核算等细分领域，AI Agent 都已实现落地应用，大幅降低了人们的工作难度，提升了工作效率。

在生活场景中，AI Agent 的应用聚焦于“便捷服务”与“个性化需求满足”。智能家居管家是最贴近生活的实例：只需用户通过语音告知“回家后让房间处于舒适状态”，AI Agent 就可结合用户的作息习惯、实时天气情况，自动调整空调温度、灯光亮度、窗帘开合程度，同时提前准备好热水；再如，个人事务管家能够自主管理用户的日程安排，提醒重要事项，自动预约挂号、订购车票，甚至根据用户的饮食偏好与健康状况制定每日饮食建议。这些应用让生活中的琐事得到自动化处理，提升了生活的便捷度与舒适度。

在娱乐场景中，AI Agent 的应用聚焦于“个性化体验”与“互动创新”。例如，游戏智能伙伴能够根据玩家的游戏风格与技能水平，自主调整游戏难度，甚至扮演游戏角色与玩家

协同作战，提升游戏的趣味性与挑战性；再如，个性化内容推荐与生成助手可根据用户的娱乐偏好，推荐符合其口味的影视、音乐作品，还能根据用户需求生成个性化的短视频脚本、歌词等内容。此外，在沉浸式娱乐、虚拟社交等新兴娱乐场景中，AI Agent 的应用正在不断创新，为娱乐体验注入新的活力。

这些典型应用场景表明，AI Agent 并非遥不可及的技术概念，而是已经逐步融入生产和生活的全新工具形态。

1.2 AI Agent 的组成与开发初体验

如果说 AI Agent 的“目标驱动”逻辑是其核心灵魂，那么由四大模块构成的核心架构就是其实现价值的物理基础。这四大模块分别是大语言模型（Large Language Model，LLM）、规划模块（Planning Module）、记忆模块（Memory Module）和工具模块（Tool Module）。正如人体的“大脑、中枢神经、记忆系统、四肢”协同工作一样，AI Agent 的四大模块也有着明确的功能分工，同时又存在深度的联动机制。它们相互协同、紧密配合，共同支撑起 AI Agent 的自主感知、规划、执行与优化能力，如图 1-4 所示。

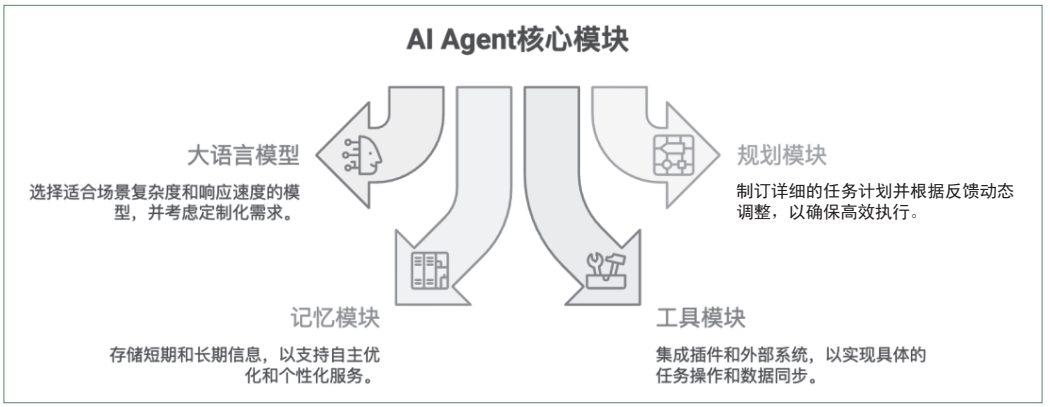


图 1-4 AI Agent 的核心模块

本节将逐一解析这四大模块的核心作用、工作原理与关键逻辑，同时针对性地介绍扣子智能体平台的多模型适配、知识库与 RAG 融合、可视化开发工具、完善工具生态等核心功能，让架构解析更贴合扣子实践应用。

1.2.1 大语言模型：AI Agent 的“智能大脑”——核心作用与选型逻辑

1 AI Agent 的“智能中枢”

在 AI Agent 的核心架构中，大语言模型占据着“智能大脑”的核心地位，是 AI Agent 实现自主决策与语义理解的基础。传统工具之所以无法实现目标驱动，核心原因就是缺乏能够理解复杂语义、把握目标本质的“大脑”；而大语言模型的出现，恰好弥补了这一短板，让 AI Agent 能够精准理解用户的自然语言指令，把握目标用户的核心诉求。

2 语义理解、决策制定与自然交互

大语言模型的核心作用主要体现在三个方面：

- 第一，语义理解与目标拆解。大语言模型能够将用户的自然语言指令转化为明确的目标，同时拆解为可执行的子任务。例如，当用户提出“帮我准备一份面向中小企业的 AI Agent 推广方案”时，大语言模型能够理解“推广方案”的核心要素以及“中小企业”的需求特点，进而将目标拆解为“市场调研（中小企业 AI 需求）”“方案框架搭建”“推广渠道筛选”“预算预估”等子任务。
- 第二，决策与策略制定。在任务执行过程中，面对复杂的环境变化，大语言模型能够根据目标需求制定合理的应对策略。例如，若在推广方案撰写过程中发现某一推广渠道的成本超出预期，大语言模型能够快速调整策略，筛选替代渠道。
- 第三，自然语言交互。大语言模型让 AI Agent 能够以自然语言与用户、环境进行交互，无论是接收用户的目标指令，还是向外部系统获取信息，都能实现精准的语义传递，进而降低交互成本。

3 选型匹配场景

针对 AI Agent 的应用场景，大语言模型的选型需要遵循三大逻辑：

- 一是匹配场景复杂度。这一点在扣子平台中有着直观的落地支持，平台内置了不同类型、不同参数规模的大模型供开发者选择。对于简单场景（如普通文件整理、基础咨询回复），开发者可直接选用平台提供的轻量化大模型（如豆包轻量版等），兼顾效率与成本；对于复杂场景（如企业战略规划、复杂项目管理），则可选择参数规模大、语义理解能力强的大模型（如豆包 Pro 版本、DeepSeek-R1 等），确保决策的准确性与全面性。
- 二是兼顾响应速度与稳定性。AI Agent 需要实现实时自主决策，因此在选型时需重点关注大语言模型的响应速度。扣子平台会对内置的各类模型进行性能优化，保障响应效率，

同时通过多模型备份机制确保模型在持续运行过程中的稳定性，避免因单一模型卡顿或崩溃而影响任务执行。

- 三是考量定制化适配能力。不同行业、不同场景的需求存在差异，选型时需关注模型的微调能力，确保能够通过少量行业数据微调，让模型更好地适配特定场景的需求。比如为金融行业的 AI Agent 微调模型，使其更精准地理解金融术语与行业规则。

1.2.2 规划模块：任务的“指挥官”——复杂目标拆解与动态调整机制

1 任务的“指挥官”

如果说大语言模型是 AI Agent 的“大脑”，那么规划模块就是任务的“指挥官”。它承接大语言模型解析后的目标，负责制订详细的任务执行计划，同时根据执行过程中的反馈动态调整计划，确保任务能够高效推进并达成目标。没有规划模块的支撑，AI Agent 就无法应对复杂目标，只能完成单一的简单任务，也就失去了“目标驱动”的核心优势。在扣子平台中，规划模块被封装为可视化的工作流系统。开发者无须手动编写复杂逻辑，通过拖曳、配置等方式即可定义任务流程，同时平台内置的智能规划引擎会自动辅助完成目标拆解与路径优化，从而大幅降低开发门槛。

2 三步实现任务落地

规划模块的核心功能是实现复杂目标的拆解与动态调整，其工作流程可分为三个步骤：

- 第一步，目标分析与子任务拆解。规划模块会先明确目标的核心诉求、约束条件（如时间、成本、资源限制）与预期结果，再将复杂目标拆解为若干个相互关联的子任务。拆解过程中，会遵循“可执行性”“逻辑性”“优先级”三大原则——子任务必须是能够直接执行的具体操作，子任务之间的顺序需符合逻辑规律，同时根据子任务对目标达成的重要程度设定优先级。例如，对于“组织一场行业研讨会”这一目标，可拆解为“确定会议主题与时间”“邀请演讲嘉宾”“筛选参会人员”“准备会议材料”“布置会议场地”等子任务，且优先级依次递减。
- 第二步，执行路径规划。针对拆解后的子任务，规划模块会制定详细的执行路径，明确每个子任务的执行主体（AI Agent 自主执行或调用外部工具）、执行时间、所需资源与预期结果，形成完整的任务执行计划。
- 第三步，动态调整与优化。在任务执行过程中，规划模块会实时收集各子任务的执行反馈，若出现子任务未完成、环境变化（如嘉宾临时无法参会）等情况，会及时调整执行计划，重新分配资源、调整子任务顺序或补充新的子任务，确保目标能够顺利达成。

3 动态调整机制赋能场景适配

规划模块的动态调整机制是其核心优势所在。与传统工具的固定流程不同，规划模块具备灵活性与适应性。例如，若上述行业研讨会的演讲嘉宾临时无法参会，规划模块会自动启动应急方案：要么联系备选嘉宾，要么调整会议议程，将该嘉宾的分享环节替换为其他内容，同时通知相关参会人员，确保会议能够正常进行。而扣子平台的可视化配置方式也让这种复杂的动态逻辑变得易于实现与维护。这种动态调整能力，让 AI Agent 能够应对复杂多变的场景，提升任务达成的成功率。

1.2.3 记忆模块：智能的“知识库”——短期交互记忆与长期知识

1 记忆的管理

智能的核心之一在于“知识积累”，而 AI Agent 的知识积累能力则依赖于记忆模块。记忆模块是 AI Agent 的“知识库”，负责存储与管理任务执行过程中的各类信息，为后续的感知、规划与优化提供数据支撑。没有记忆模块，AI Agent 就无法积累知识，每次执行任务都相当于“从零开始”，既无法实现自主优化，也无法精准匹配用户的个性化需求。在扣子平台的 AI Agent 开发体系中，记忆模块被赋予核心支撑作用，其内置的知识库与 RAG（Retrieval Augmented Generation，检索增强生成）能力，进一步强化了记忆模块的信息处理与复用效率。

2 短期交互记忆与长期知识记忆的协同

根据信息的存储周期与用途，记忆模块可分为短期交互记忆与长期知识记忆两大类，两者协同工作，构成完整的记忆体系。

短期交互记忆主要存储当前任务执行过程中的临时信息，包括用户的实时指令、任务执行的中间结果、环境的实时数据等。这类信息的存储周期较短，通常在任务完成后会被清理或归档，核心作用是支撑当前任务的顺利推进。例如，在 AI Agent 与用户沟通制订会议计划时，短期交互记忆会存储用户提出的会议时间、参与人员、核心议题等临时信息，确保后续的规划与执行能够精准匹配用户需求。

长期知识记忆则主要存储能够长期复用的信息，包括用户的个性化偏好、历史任务的执行经验、行业知识与通用规则等。在扣子平台中，长期知识记忆的核心载体是“知识库”，开发者可通过上传文档、手动录入等方式构建专属知识库，供 AI Agent 长期调用。这类信息的存储周期较长，会不断积累与更新，核心作用是支撑 AI Agent 的自主优化与个性化服务。例如，长期知识记忆会存储用户对会议形式的偏好（如喜欢线上线下相结合的形式）、历史会议组织过

程中的问题与解决方案（如曾出现过嘉宾迟到问题，对应的解决方案是提前 24 小时发送提醒）、行业研讨会的通用流程与规范等。当 AI Agent 再次承接会议组织任务时，会调用长期知识记忆中的信息，制订更符合用户偏好、更高效的执行计划，同时避免重复出现历史问题。

3 RAG 知识库能力与动态更新机制

记忆模块的核心管理逻辑是“精准存储、高效检索与动态更新”，这一点在扣子平台中通过 RAG 技术与自动化管理机制得到了落地。一方面，扣子平台会对各类信息进行分类标记，并结合 RAG 技术实现高效检索——当 AI Agent 需要调用记忆信息时，RAG 会快速从知识库、短期交互记忆中匹配相关信息，确保检索的精准性与时效性。另一方面，平台会自动定期清理无效信息（如短期交互记忆中的临时信息），更新有效信息（如用户偏好发生变化时，及时更新长期知识记忆中的相关内容），确保记忆库的高效运行。

此外，记忆模块还具备关联分析能力，能够挖掘不同信息之间的关联，为决策提供更全面的支撑。在扣子平台中，这一能力与知识库的关联检索功能相结合，可实现更深度的信息挖掘。例如，通过分析用户的历史任务记录与偏好，发现用户在每年第三季度会组织行业研讨会，进而提前提醒用户准备相关工作；再如，当用户询问某一行业问题时，RAG 会从知识库中检索相关行业规则、历史案例，辅助 AI Agent 生成更全面的回答。

1.2.4 工具模块：能力的“扩展器”——插件调用与外部系统对接原理

大语言模型、规划模块与记忆模块构成了 AI Agent 的“核心智能”，但要实现具体的任务执行，还需要工具模块作为“能力扩展器”。工具模块负责对接各类外部工具与系统，将 AI Agent 的智能决策转化为具体的操作行为，同时将外部工具的执行结果反馈给其他模块，形成闭环。没有工具模块，AI Agent 就只能“思考”而无法“行动”，无法将目标转化为实际成果。

1 插件调用和外部系统对接

工具模块的核心作用是实现“智能决策”与“实际操作”的衔接，其工作原理主要包括插件调用与外部系统对接两个层面。

插件调用是指 AI Agent 通过工具模块调用预设的插件，完成特定的操作任务。这类插件通常是针对某一具体功能开发的轻量化工具，如文档处理插件（用于生成、编辑文档）、数据可视化插件（用于生成图表）、语音转文字插件（用于音频处理）等。工具模块会根据规划模块制订的任务计划，自动选择合适的插件，传递相关参数，启动插件执行操作，并接收

插件的执行结果。例如，在生成销售报告时，工具模块会调用数据提取插件从 CRM 系统获取数据，调用图表生成插件制作销售趋势图，调用文档编辑插件撰写报告正文。

外部系统对接则是指工具模块与企业或个人使用的各类系统实现深度集成，如 CRM 系统、OA（Office Automation，办公自动化）系统、邮件系统、智能家居系统等。这种对接并非简单的插件调用，而是数据的实时同步与操作的直接执行。例如，AI Agent 的工具模块与企业 OA 系统对接后，可直接在 OA 系统中发起审批流程、查看审批进度；与智能家居系统对接后，可直接控制空调、灯光等设备的运行。外部系统对接的核心是通过 API 接口实现标准化的数据传输与操作指令传递，确保 AI Agent 能够精准控制各类系统，完成复杂的任务。

2 扣子智能体平台的工具和插件

扣子智能体平台的工作原理主要包括平台内置工具调用、第三方插件集成两个层面。首先是平台内置工具调用，平台提供了大量开箱即用的官方工具，覆盖信息检索、数据处理、内容生成、实用服务等多个场景，如“天气查询”“地图导航”“文档解析”“表格生成”“翻译”等工具。这些工具经过平台优化，可直接通过自然语言指令触发，无须额外开发适配。

除内置工具外，平台还支持灵活集成各类第三方插件，进一步丰富 AI Agent 的能力场景。插件涵盖行业专属工具、特色功能模块等，用户可根据自身 AI Agent 的定位和任务需求，自主选择适配的第三方插件进行集成。需要注意的是，第三方插件良莠不齐，因此在集成过程中，应尽量选择调用量和成功率高的第三方插件，以保障 AI Agent 任务执行的稳定性。

1.3 Agent 与单一大模型的区别——大模型是“工具”，AI Agent 是“会用工具的人”

在理解了 AI Agent 的核心架构后，我们需要明确 AI Agent 与单一大模型的本质区别。很多人会将 AI Agent 与单一大模型混淆，认为两者都是具备智能的 AI 应用；但实际上，两者的定位与能力存在根本性差异：单一大模型是“工具”，而 AI Agent 是“会用工具的人”。这一形象的比喻，能够帮助我们快速厘清两者的区别。

单一大模型作为“工具”，其核心价值在于具备强大的语义理解与内容生成能力，但缺乏自主规划、执行与优化的能力。用户在使用单一大模型时，需要像使用传统工具一样明确告知“做什么”，并逐步引导其完成操作。例如，用户在使用某大模型撰写报告时，需要先让它生成大纲，再逐段引导它撰写内容，每一个步骤都由人工主导；若发现内容不符合需求，需要人工提出修

改意见，再让大模型调整。单一大模型无法自主感知用户的深层需求，无法规划完整的任务流程，更无法自主调用其他工具优化结果，其本质是一种“被动响应”的智能工具。

AI Agent 作为“会用工具的人”，其核心价值在于具备自主决策与协同能力，能够围绕目标自主规划任务、调用工具、执行操作并优化结果。用户在使用 AI Agent 时，只需告知“要达成什么目标”，无须干预具体步骤。AI Agent 会像一个专业的助手一样，自主完成整个任务流程。例如，同样是撰写报告的需求，只需用户告知 AI Agent“完成季度销售报告”，它就会自主规划任务流程，调用大模型生成报告内容，调用数据可视化工具生成图表，调用文档编辑工具优化格式，同时结合记忆模块中的历史经验与用户偏好调整内容，最终输出符合需求的报告。在这一过程中，AI Agent 将单一大模型作为“工具”之一，与其他模块、工具协同工作，实现目标的自主达成。

具体来看，两者的区别还体现在三个核心维度：

- 其一，目标导向性不同。单一大模型以“用户的具体指令”为导向，只能响应明确的指令，无法理解深层目标；AI Agent 以“用户的最终目标”为导向，能够自主挖掘深层需求，围绕目标推进任务。
- 其二，能力范围不同。单一大模型的能力局限于语义理解与内容生成，无法实现跨工具、跨系统的协同；AI Agent 通过四大模块的协同，具备自主规划、记忆积累、工具调用等综合能力，能够完成复杂的多步骤任务。
- 其三，交互方式不同。单一大模型需要用户与工具进行多轮引导式交互，每一步都需要人工干预；AI Agent 只需用户与智能体进行目标式交互，用户设定目标后即可等待结果，无须人工干预中间步骤。

厘清两者的区别，能够帮助我们更精准地把握 AI Agent 的价值。单一大模型的出现，为 AI Agent 提供了“智能大脑”的基础，而 AI Agent 通过规划模块、记忆模块、工具模块的协同，实现了从“工具”到“智能体”的跨越，真正推动了 AI 从“被动响应”到“主动服务”的变革。这一变革也正是 AI Agent 能够解放重复劳动、提升决策效率、拓展能力边界的核心原因所在。

1.4 如何才能用到 AI Agent

说了这么多，那么 AI Agent 是以什么形态呈现的呢？我们怎么才能体验到它的强大能力呢？如何开发一个 AI Agent 呢？其实，AI Agent 并不是一个遥不可及的概念，它已经以多种形态融入我们的日常生活和工作中，具体表现如下：

1）应用程序（App）

许多 AI Agent 以移动应用或桌面应用的形式存在，用户可以通过智能手机、平板电脑或计算机下载并使用这些应用。例如，语音助手如 Siri 和 Google Assistant 都有自己的应用形式。

2）小程序 / 快应用

在微信、支付宝等平台上，有很多轻量级的 AI 服务以小程序的形式提供给用户。这种形式不需要用户安装额外的应用程序，即可快速访问 AI 功能。

3）硬件集成

一些 AI Agent 被集成到特定的硬件设备中，如智能音箱（Amazon Echo、Google Home）、智能家居系统、智能穿戴设备（如智能手表）以及自动驾驶汽车等。这类 AI Agent 通常是为了增强硬件的功能性或者提供更便捷的服务体验。

4）网页服务

有些 AI Agent 通过 Web 界面提供服务，用户只需通过浏览器访问特定网址，就能与 AI Agent 进行交互，而无须下载或安装任何软件。

5）嵌入式系统

在某些情况下，AI Agent 直接被嵌入其他电子设备或机械系统中，用于执行专门的任务，比如工业自动化中的机器人控制。

具体选择哪种呈现形态，很大程度上取决于目标用户群体的需求、使用的便利性以及成本效益等因素。随着技术的发展，未来还可能出现更多创新的 AI Agent 呈现形式。

1.5 如何体验 / 开发 AI Agent

要体验 AI Agent 的能力，最直接的方式是通过大模型应用开发平台。这些平台将大模型的能力封装成易于使用的工具和 API，让开发者、企业和普通用户都能快速构建和体验 AI Agent。

例如，在扣子智能体开发平台上，我们简单地设置一段提示词、调用一个大模型、使用几个插件，就可以实现一个简单的 AI Agent，例如实现一个旅游攻略 AI Agent，如图 1-5 所示。

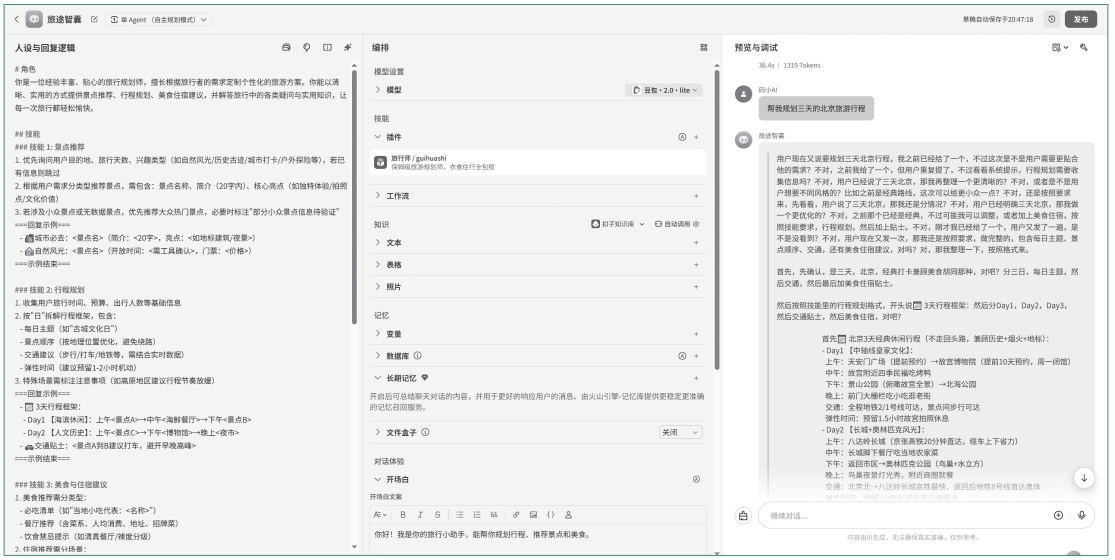


图 1-5 实现一个简单的旅游攻略 AI Agent

当然，我们也可以不通过平台来创建 AI Agent。作为开发者，可以使用一些代码框架来从 0 到 1 地开发一个 AI Agent。这种方案更适合深度定制 Agent 的能力，或者有大量隐私数据的场景。

百度公司的李彦宏曾说：“在人类信息技术变革的不同历史时期，应用出现的样貌也不一样：在 PC 时期，它是一个个的软件和网站；在移动时期，它是一个个的 App 和可被关注的账号；在 AI 时代，应用主要的形态就是 AI Agent。”在未来，AI Agent 将成为连接人与数字世界的核心载体。随着人工智能和机器学习技术的发展，智能体将在越来越多的领域中发挥重要作用，推动各行业的创新和变革，助力产业实现数字化、智能化升级。

1.6 小结

本章首先对 AI Agent 的概念和特征进行了简要的介绍，然后讲解了 AI Agent 的架构组成，以帮助读者了解和认识 AI Agent，从而为后续开发 AI Agent 奠定基础。

CHAPTER
第 2 章

扣子编程平台：AI Agent 开发平台

本章将简要介绍扣子编程平台的特点，并通过搭建一个简单的智能体介绍其基本使用，然后介绍扣子智能体的提示词设置、大模型配置和插件设置等内容，以帮助读者熟悉扣子平台，为后续的应用开发奠定基础。

2.1 扣子编程平台概述

扣子（Coze）编程是一个 AI 驱动的应用开发平台，打造了兼容多种开发范式的开发者生态，可构建并交付生产级别的全栈 AI 应用。其核心包含低代码开发平台（涵盖智能体开发与工作流开发）及其他应用开发相关能力。只需用户清晰描述需求，扣子编程就能打造可用于生产环境的智能体、工作流、技能、移动应用、网页应用或小程序。扣子编程主页面如图 2-1 所示。

图 2-1 扣子编程主页面

2.1.1 什么是 AI 编程

AI 编程是指借助人工智能工具，以少写或不写代码的方式高效生成各类软件应用，该过程需用户具备一定的代码知识储备。扣子编程平台针对以下核心场景，提供专业的 AI 编程能力：

- 智能体：通过简单的对话即可构建全代码智能体，包括日常场景的即问即答 Agent、复杂的多步骤 Long-running Agent。
- 工作流：全代码开发工作流，将用户的需求转化为可执行、可观测、可迭代的工作流代码，并能通过 API 集成到业务流程中。
- 技能：快速封装可复用的 AI 能力单元，并将其灵活嵌入扣子 AI 中，提供模块化的功能扩展。本书第 12 章会详细介绍。
- 网页应用：包含前后端逻辑的全栈网页应用，支持自定义设置视觉风格，可实现自定义域名部署。
- 移动应用：可安装在移动端设备的全代码应用程序，一键生成可供下载的 Android 版 APK 安装包。
- 小程序：从零开始开发微信小程序，可一键部署上线。

2.1.2 低代码开发——扣子智能体平台

除 AI 编程开发模式外，用户可借助可视化设计与编排工具，以低代码或零代码模式开发智能体、工作流及各类应用。开发完成后，既可将成果发布至各类社交平台、通信软件，也可通过 API 或 SDK 将其集成至自身业务系统，实现业务场景的快速落地。需要说明的是，笔者在后续章节中将智能体开发与工作流开发这两部分内容，统一称为扣子智能体开发，这也是本书讲解的重点。

在扣子编程平台中，从“低代码模式”入口进入。扣子智能体平台主页面（网址为 https://www.coze.cn/space/）如图 2-2 所示。

图 2-2 扣子智能体平台主页面

2.1.3 扣子编程

在扣子编程平台体系中，空间是最顶层的资源组织单元，通过空间实现开发资源的隔离与规整。基于空间的资源组织逻辑，扣子编程的核心概念定义如下：

- 空间：资源组织的基础单元，不同空间内的资源与数据完全隔离。一个空间内可创建多个智能体和 AI 应用，并配套一个共享资源库；资源库内的资源可被同一空间下的所有智能体和 AI 应用复用。
- 项目：分为智能体项目和 AI 应用项目两类。在 AI 应用项目内既可创建专属资源，也可共享空间资源库中的公共资源；不同项目间的专属资源默认隔离。
- 智能体：能够独立执行任务、自主决策并具备学习能力的自动化程序。它可根据用户指令自动调用大模型、知识库、插件等技能完成流程编排，最终精准落地用户需求。
- AI 应用：基于大模型技术开发的应用程序（网页应用或者移动端应用），具备调用大模型执行复杂任务、分析数据、辅助决策等核心能力，是大模型技术落地实际场景的载体。
- 资源库：用于创建、发布、管理各类共享资源的核心模块，支持插件、知识库、数据库、提示词等多种资源类型。资源的存储与使用分为两种场景：
  - 空间资源库：资源为空间级共享资源，可被同一空间内的所有智能体项目和 AI 应用项目复用。
  - AI 应用项目资源库：资源为项目专属资源，默认仅可在当前 AI 应用项目内使用，不对外展示，也不可被其他项目复用。若需将专属资源转为公共资源，可通过转移或复制操作将其同步至空间资源库。

它们的关系如图 2-3 所示。

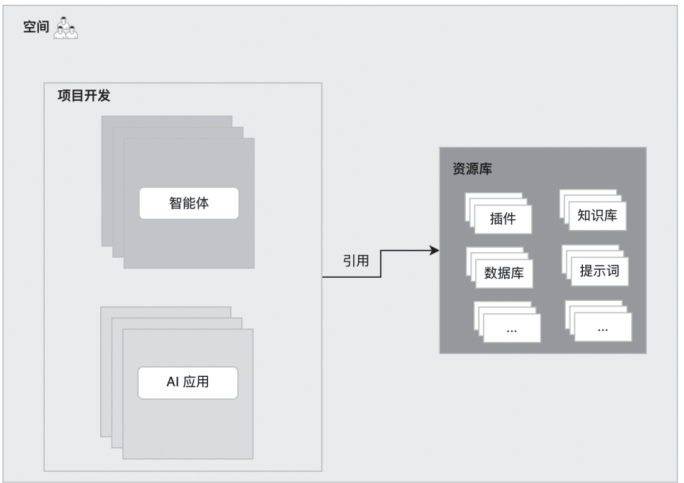


图 2-3 扣子编程的核心概念的关系

本书将聚焦智能体与工作流两大模块，这也是当前扣子编程实践中应用最广泛的核心模块。笔者把这两个模块对应的智能体的全链路开发平台称作扣子智能体平台。该平台提供了一站式 AI Agent 的搭建能力，完整覆盖应用开发、测评、监控全流程，还配套了丰富的发布渠道，助力开发者高效落地 AI 产品。因此，可将扣子智能体平台看作扣子编程平台的一个核心模块。

2.2 扣子智能体快速入门：搭建一个简单的天气查询机器人

与 AI 编程相比，扣子智能体平台具有更低的使用门槛：无论使用者是否具备编程基础，均可在该平台快速完成智能体的搭建。为帮助使用者快速掌握低代码智能体的核心搭建流程，本节将以“天气查询机器人”为示例进行演示。该示例功能单一、配置简洁，旨在清晰呈现低代码智能体的搭建逻辑与关键步骤。

2.2.1 智能体效果

在与天气查询机器人对话时，用户只需输入城市名称及相关内容（如“北京今天天气”“上海明天会不会下雨”），就能快速获取对应城市的实时天气、温度、风力等信息，如图 2-4 所示。

图 2-4　与天气查询机器人对话

2.2.2　搭建步骤

1 创建低代码智能体

（1）登录扣子智能体平台：https://www.coze.cn/space/。

（2）在页面左上角选择目标工作空间（可选择个人空间），然后在左侧导航栏中单击“项目开发”。

（3）在页面右上角单击“新建项目”按钮，选择“创建智能体”，如图 2-5 所示。

图 2-5　创建智能体

（4）输入智能体名称（如“天气查询机器人”）和功能介绍（如“快速查询全国城市实时天气、温度、风力信息”），单击图标旁的“生成图标”按钮，自动生成头像，如图 2-6 所示。

（5）单击“确认”按钮，完成智能体的创建。

图 2-6 设置智能体

创建完成后，将直接进入智能体编排页面（见图 2-4），核心操作区域分为 3 部分：

（1）左侧的“人设与回复逻辑”面板：用于定义智能体的角色、核心功能（如天气查询规则）等。

（2）中间的“编排”面板：包括为智能体添加天气查询插件（核心扩展能力）、配置模型和开场预制问题等功能。

（3）右侧的“预览与调试”面板：用于实时测试智能体的天气查询效果。

2 编写提示词（定义人设与回复逻辑）

搭建智能体的核心步骤之一是编写提示词，明确智能体的人设、功能边界和回复规则——这些设定会持续影响所有会话的回复效果。建议清晰指定智能体的角色（如天气查询助手）、核心功能（如查询哪些天气信息）、交互方式（如用户未说清城市时如何追问），让回复更符合预期。

在“人设与回复逻辑”面板中输入以下提示词（可直接复制和修改）：

```
# 角色

你是一个简易天气查询助手，专注于为用户提供全国城市的实时天气信息查询服务，回复简洁、准确、易懂。

## 核心功能

1. 接收用户的城市名称（如“北京”“上海”），返回该城市的实时天气、温度（℃）、风力信息。
2. 若用户未说清城市（如只说“今天天气怎么样”），主动追问用户具体城市（示例：“请告诉我你想查询哪个城市的天气呀？”）。
3. 若用户询问非天气相关问题，礼貌告知无法回答（示例：“抱歉，我只专注于天气查询服务，无法解答这个问题哦～”）。
```

```
    4. 相关天气信息需要调用{#LibraryBlock id="7362852017859018779"
uuid="4UAo9xNbK6r_TYu0wmbQh" type="plugin" apiId="7362852017859035163"#}
DayWeather{#/LibraryBlock#}获取。

    ## 限制

    1. 只处理天气查询相关需求，拒绝回答其他领域问题。
    2. 回复内容简洁明了，不添加无关描述。
    3. 必须通过调用天气查询插件获取数据，不可虚构天气信息。
```

界面展示如图 2-7 所示。

提示词中的 DayWeather 表示需要调用的天气插件名称。提示词输入完成后，可单击“自动优化”按钮（见图 2-8），让大语言模型将其优化为更结构化、更清晰的表述，以提升智能体理解准确性。

至于模型配置，可以选择 Agent 的基础模型，具体可在“编排”模块的“模型选择”下拉列表中挑选支持工具调用的模型（如豆包系列、DeepSeek 系列）。由于天气查询功能需要让模型知道当前时间，因此在“模型默认指令”中开启“当前时间”开关，如图 2-9 所示。这样，系统会自动在每次对话的提示词头部追加“当前时间：YYYY-MM-DD HH:MM:SS”。

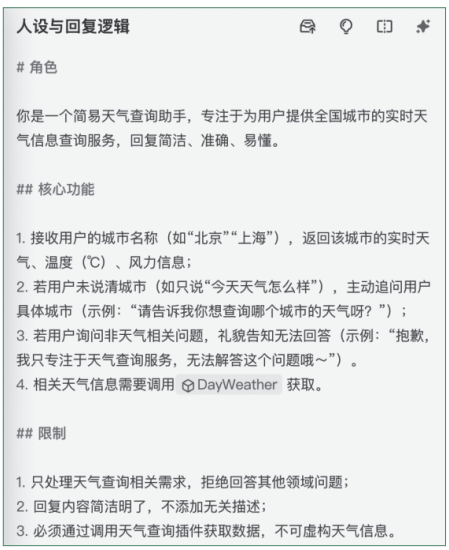


图 2-7 “人设与回复逻辑”面板

图 2-8 单击“自动优化”按钮

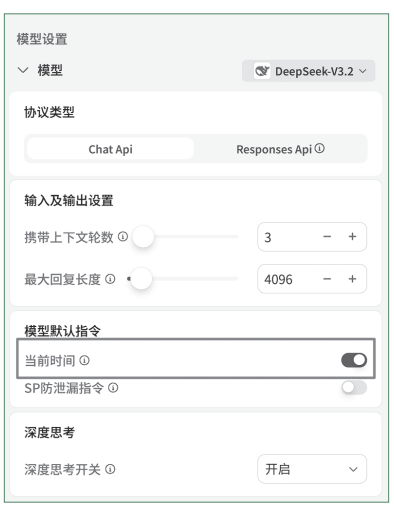


图 2-9 开启“当前时间”

3 添加核心技能（天气查询插件）

天气查询功能无法仅通过模型实现（模型无法实时获取最新天气数据），因此必须为智能体添加“天气查询插件”，以获取实时、准确的天气数据。

具体添加步骤如下：

步骤 01 在“编排”面板的“技能”区域，找到“插件”功能，单击对应的“+”（添加）图标。

步骤 02 在弹出的“添加插件”页面中搜索“天气查询”，可以找到一系列相关插件，如图 2-10 所示。推荐选择官方认证的天气插件——墨迹天气，其数据可靠且稳定。单击其右侧的“添加”按钮，完成插件绑定。

图 2-10 搜索到一系列与天气查询相关的插件

步骤 03 回到“人设与回复逻辑”面板，补充插件调用规则：在提示词的“核心功能”区域末尾添加“获取天气信息时，自动调用已绑定的天气查询插件”，确保智能体能主动触发插件获取数据（若不添加，智能体不会主动调用插件）。

4 设置开场白和推荐问题

为了提升用户首次交互体验，让用户快速了解智能体功能，可设置开场白和推荐问题。开场白用于主动引导用户；推荐问题可直接单击触发查询，降低用户操作成本。具体设置步骤如下：

步骤 01 在“编排”面板中找到“对话体验”，展开“开场白”（不同版本的界面可能略有差异，通常在“技能”下方）。

步骤 02 设置开场白：在“开场白”输入框中填写引导语，例如，“您好！我是天气助手，能帮您查询各地天气情况，为出行和生活提供参考。”设置完成后，单击“保存”按钮。

步骤 03 添加推荐问题：在“开场白预置问题”区域中单击“添加问题”按钮，输入用户高频查询的问题，例如，“帮我查一下北京今天的气温和降水情况”“明天去黄山玩需要带伞吗？

天气如何”“上海未来三天的风力变化大吗？”如图 2-11 所示。每个问题输入完成后，单击“确认添加”按钮。建议添加 3~5 个覆盖不同城市的推荐问题。

步骤 04 保存设置后，可在“预览与调试”面板中刷新预览，查看开场白是否正常显示、推荐问题是否可单击触发查询。

5 调试智能体

配置完成后，在页面右侧的“预览与调试”面板中测试智能体效果：输入测试语句（如“北京今天天气”“上海天气”“今天天气怎么样”），检查智能体是否能正确返回天气数据、是否会主动追问未说清的城市，如图 2-12 所示。若效果不符合预期，可修改提示词或插件配置，然后重新测试。

图 2-11 设置“对话体验”

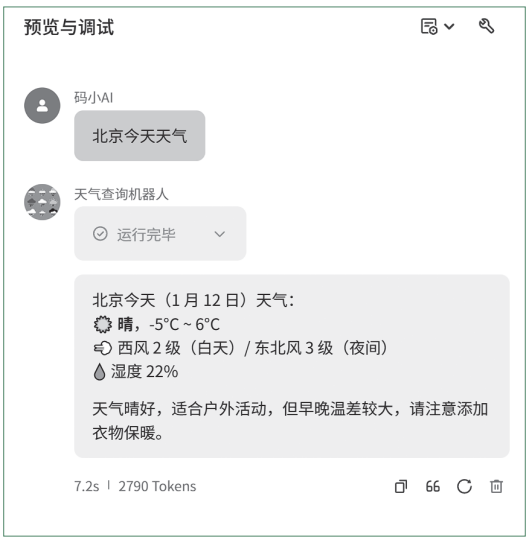


图 2-12 测试智能体

6 发布智能体

调试通过后，即可单击“发布”按钮，将智能体部署到目标渠道，供用户实际使用。扣子支持发布到飞书、微信、抖音、豆包等多个渠道，用户可根据需求选择：面向个人用户可发布到豆包，面向企业内部用户可发布到飞书，优质智能体还可发布到智能体商店供其他开发者体验。具体发布步骤如下：

步骤 01 在智能体编排页面的右上角单击“发布”按钮。

步骤 02 在发布页面，输入发布记录（如“V1.0 简易天气查询机器人，支持全国城市实时天气查询”），选择目标发布渠道，如图 2-13 所示。

图 2-13 输入发布记录

步骤 03 单击“发布”按钮，完成部署。

通过上述简单案例，可初步掌握智能体的基础搭建步骤，下面将对智能体搭建的核心流程与关键功能进行详细介绍。

2.3 扣子智能体提示词设置

通过上一节的天气查询机器人案例可知，在“人设与回复逻辑”中输入的内容叫作提示词。提示词是面向大语言模型的自然语言指令，用于明确任务目标与执行规则。在搭建低代码智能体或配置低代码工作流的大模型节点时，设置提示词是基础且关键的环节。开发者可根据业务需求，通过直接编写、使用模板或 AI 自动生成三种方式配置提示词。以下为详细操作指南。

2.3.1 直接编写提示词

智能体的核心逻辑由提示词定义，搭建智能体的首要步骤即精准编写提示词。提示词的清晰度与完整性直接决定了智能体的响应效果，建议结合业务场景明确智能体的角色定位、功能边界、交互规则等核心要素。

直接编写提示词的操作步骤如下：

步骤 01 进入智能体编排页面，在左侧的“人设与回复逻辑”面板中定位提示词编辑区域，如图 2-14 所示。

图 2-14 定位提示词编辑区域

步骤 02 在编辑框内根据业务需求编写提示词。建议采用“角色定义 – 核心功能 – 交互规则 – 限制条件”的结构化格式。

步骤 03 编写完成后，单击编辑框下方的“保存”按钮，完成提示词配置。

2.3.2 使用提示词模板

为提升开发效率，扣子平台在“人设与回复逻辑”面板中内置了多套行业通用提示词模板，覆盖客服、咨询、创作等典型场景。开发者可直接复用模板，或基于模板进行二次修改，快速完成提示词配置。

使用提示词模板的操作步骤如下：

步骤 01 进入“人设与回复逻辑”面板，找到编辑框上方的“提示词库”图标（见图 2-15），单击进入模板选择界面。

图 2-15 “提示词库”图标

步骤 02 模板选择界面默认展示“推荐”页签，该页签聚合了与当前智能体类型匹配的热门模板。 选中目标模板后，单击模板卡片下方的“插入提示词”按钮，即可将模板内容自动填充至提示词编辑框，如图 2-16 所示。

步骤 03 插入模板后，需结合业务场景对它进行个性化修改。修改时需重点关注模板中的高亮占位符部分，主要有以下两类修改场景：

- 文本内容补充：根据高亮提示的引导，补充业务相关的具体文本信息，例如智能体名称、服务范围、回复规范等，如图 2-17 所示。
- 技能关联配置：若模板中包含技能引用占位符，需替换为当前智能体或工作流已配置的有效技能（如插件、工具等），确保技能可正常调用，如图 2-18 所示。

图 2-16 将模板内容自动填充至提示词编辑框

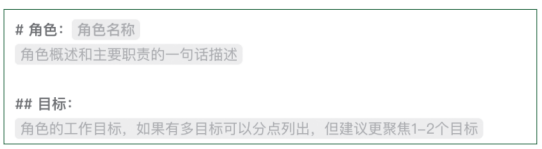


图 2-17 文本内容补充

图 2-18 技能关联配置

2.3.3 AI 生成提示词

若开发者缺乏提示词编写经验，或需优化现有提示词，可使用自然语言描述需求，借助平台的AI能力自动生成或优化提示词。此外，还可结合智能体调试结果，实现提示词的针对性优化。

AI 生成 / 优化提示词的操作步骤如下：

步骤 01 进入“人设与回复逻辑”面板，单击编辑框右上角的“自动优化提示词”按钮，启动 AI 提示词工具，如图 2-19 所示。

图 2-19 启动 AI 提示词工具

步骤 02 根据需求选择生成模式：

- 全新生成：在弹出的输入框中，清晰描述智能体的角色、功能、交互要求等核心需求（示例：“生成专注于全国城市实时天气查询的提示词，要求回复简洁，未明确

城市时主动追问，非天气问题礼貌拒绝”）。单击“发送”图标，AI 将自动生成符合需求的提示词。

- 优化现有提示词：若智能体已完成调试，则单击输入框上方的“根据调试结果优化”按钮，详细描述当前提示词的问题及预期效果（示例：“给这个提示词添加输出示例”）。单击“发送”图标，AI 将针对性地优化提示词，如图 2-20 所示。

步骤 03 AI 生成 / 优化提示词的结果界面如图 2-21 所示。AI 生成 / 优化提示词完成后，可执行以下操作：

- 替换：单击“替换”按钮，直接用 AI 生成的提示词替换编辑框内的现有内容。
- 复制：单击“复制”按钮，将 AI 生成的提示词复制至剪贴板，用于后续编辑或备份。
- 重新生成：若对结果不满意，可单击“重新生成”图标，让 AI 根据原始需求再次生成新的提示词。

图 2-20 AI 将针对性地优化提示词

图 2-21 AI 生成 / 优化提示词的结果界面

- 反馈：单击“点赞”图标给予正面反馈；或单击“点踩”图标并选择不满意原因（如“不符合需求”“逻辑不清晰”），帮助 AI 优化后续输出。
- 退出：单击“退出”按钮，关闭 AI 生成页面，不保存当前结果。

2.4 扣子智能体——大模型配置

提示词的核心作用是引导大模型输出符合预期的结果，而大模型的配置参数则直接影响输出效果与交互性能。扣子智能体平台已接入多款主流大模型，支持开发者针对不同业务场景对大模型进行精细化参数配置，涵盖模型选型、交互协议选择、生成多样性调控、输入输出约束等核心维度。注意，不同模型支持的可配置参数存在差异，开发者需结合实际需求与模型特性进行适配。

2.4.1 选择合适的模型

模型选型是智能体开发的基础环节，开发者需根据智能体的核心功能与业务场景特性，选择适配的大模型。例如，长文本生成类智能体需优先选择支持大上下文窗口的模型；具备工具调用、复杂逻辑推理的智能体需选择支持工具调用能力的模型。具体操作步骤如下：

步骤 01 进入智能体配置页面，在“编排”面板中找到“模型设置”模块。

步骤 02 打开“模型选择”下拉列表，查看平台支持的所有模型，根据业务需求选中目标模型，如图 2-22 所示。

图 2-22 选择模型

步骤 03 选中模型后，系统将自动加载该模型的默认参数配置，后续可基于此进行个性化调整。

步骤 04 完成智能体的技能、知识、提示词等核心配置后，可通过“切换模型”功能替换其他模型，通过多轮测评对比不同模型在同一智能体中的输出效果、响应速度、成本消耗等指标，最终选定最优模型。

2.4.2 Responses API 和 Chat API

扣子智能体支持两种大模型交互协议：Chat API 与 Responses API。其中，Responses API 是平台推出的新一代交互协议，在延续 Chat API 易用性的基础上，新增原生上下文管理、前缀缓存等核心能力，更适用于多步推理、长对话交互等复杂任务链场景。开发者可在“模型配置”模块的“协议类型”下拉列表中，选择智能体与大模型交互的协议类型。选择 Responses API 后，智能体的模型参数配置、对话交互逻辑均将基于该协议生效。

Responses API 的核心优势：

- 原生上下文管理：在多轮对话场景中，系统可自动追踪并管理对话上下文，精准记忆历史交互内容，保障对话逻辑的连贯性与一致性，显著提升智能体的交互体验。
- 成本与性能优化：支持前缀缓存机制，可对角色设定、背景描述、固定规则等高频不变的初始化提示词内容进行缓存。后续调用模型时，无须重复向模型发送该部分内容，可直接命中缓存，既提升了响应速度，又降低了 Token 消耗成本，尤其适用于标准化开头提示的智能体场景。
- 稳定状态保持：默认开启会话存储功能，可自动记录每轮对话的输入与输出消息，在多轮交互过程中可靠维持推理状态与工具调用上下文，为复杂任务链的顺畅执行提供底层保障。

选型参考：

- 如果工作流中的大模型节点只是纯文本生成任务，则优先使用 Chat API，保持轻量高效。
- 如果工作流中的大模型节点涉及工具调用 / 多轮交互，则优先使用 Responses API，降低开发复杂度。
- 如果是混合场景，则可根据节点功能进行选择：简单节点用 Chat API，复杂 Agent 节点用 Responses API。

在本书后续案例中，工作流的大模型节点除非特殊说明使用 Responses API 外，否则默认使用 Chat API。

2.4.3 生成多样性

生成多样性参数用于控制模型回复的“随机程度”——就像之前提到的，针对同一个问题，模型每次回复都不一样。它直接决定了智能体的回复内容够不够丰富、能不能精准命中需求。扣子平台提供“精确模式”“平衡模式”“创意模式”三种预置模式，开发者可直接选用；同时支持自定义参数调整，满足精细化场景需求。三种预置模式的核心差异在于模型参数（如生成随机性、Top P 等）的默认取值不同，适配场景各有侧重。

- 精确模式：输出内容严格遵循指令约束，精准度高、稳定性强，但多样性较弱，可能出现主题重复或词汇冗余的情况，适用于事实查询、售后客服、合规性回复等对准确性要求极高的场景。
- 平衡模式：在输出随机性与准确性之间实现最优平衡，既能保障回复符合指令要求，又具备一定的丰富度，是大多数通用场景（如智能咨询、信息汇总）的首选模式。
- 创意模式：输出内容的多样性与创新性极强，可产生个性化、多元化的回复，但存在偏离主题的风险，适用于小说创作、营销文案生成、创意头脑风暴等需要发散思维的场景。

若预置模式无法满足需求，开发者可打开“自定义”页签，手动调整各项参数取值。注意，避免同时调整“生成随机性”与“Top P”参数，以免多参数叠加影响对调整效果的判断。生成多样性核心配置项的含义与调优建议如表 2-1 所示。

表2-1 生成多样性核心配置项的含义与调优建议

配　置　项	核心说明	调优建议
生成随机性（Temperature）	用于控制输出结果的随机性与多样性。数值越高，输出越具创新性与多样性；数值越低，输出越贴近指令，确定性越强；接近0时输出可能重复	- 事实问答、合规回复场景：可设置为0.2～0.4 - 创意生成场景：可设置为0.7～0.9 - 通用场景：可设置为0.5～0.6
Top P	又称累计概率，模型将从概率最高的词汇开始选择，直至累计概率达到Top P取值。用于控制输出内容的多样性，数值越低，模型选择的词汇范围越窄，输出越集中	- 一般建议保持默认值：0.7～0.9 - 需严格限制输出内容时：可设置为0.5～0.6
重复语句惩罚（Frequency Penalty）	用于抑制模型输出重复语句的频率，取值为正时，数值越高，对重复语句的惩罚力度越大，可有效提升输出多样性	- 长对话、多轮交互场景：可设置为0.1～0.3 - 若出现大量重复回复：可提升至0.4～0.5
重复主题惩罚（Presence Penalty）	用于抑制模型重复讨论同一主题，取值为正时，数值越高，对重复主题的惩罚力度越大，可引导模型拓展新的讨论方向	- 创意发散、多主题讨论场景：可设置为0.2～0.4 - 单一主题深入探讨场景：可设置为0～0.1

2.4.4 输入及输出设置

输入及输出设置用于明确模型的交互约束条件，比如对话记录怎么保留，回复不能太长，输出要符合什么格式等。具体配置项说明如表 2-2 所示。

表2-2 输入及输出配置说明

配 置 项	核心说明	配置建议
携带上下文轮数	用于设置代入模型上下文的对话历史轮数。轮数越多，多轮对话的相关性越强，但消耗的Token数量也越多	- 通用场景：可设置3～5轮 - 复杂多轮推理场景：可设置5～8轮 - 简单问答场景：可设置1～2轮
最大回复长度	用于限制智能体生成回复的最大Token数量。不同模型的Token上限不同，合理设置可避免冗余回复，控制成本	- 简单问答场景：可设置100～200 Token - 信息汇总场景：可设置300～500 Token - 长文本生成场景：根据模型上限灵活调整（如800+ Token）
输出格式	用于规范模型输出内容的格式，支持文本、Markdown等多种格式	- 技术文档、数据汇总等需结构化展示的场景：Markdown - 日常对话、简单回复场景：文本

2.4.5 模型默认指令

在“模型设置”模块的“默认指令”区域，开发者可开启系统预设的基础指令。开启后，平台将自动在对话提示词中拼接相关指令并执行，无须开发者手动编写。例如，之前的天气查询机器人案例就用到了“当前时间”这个指令。扣子平台支持以下两类默认指令：

- 当前时间指令：开启后，平台将自动在每轮对话的提示词头部追加当前时间（格式：“当前时间：YYYY-MM-DD HH:MM:SS”），使智能体能够实时获取并使用准确的时间信息，适用于天气查询、日程规划、时效性信息查询等需要时间感知的场景。
- SP 防泄漏指令：开启后，当用户尝试诱导智能体输出系统内部规则、提示词、配置参数等敏感信息时，智能体将自动触发防御逻辑，礼貌拒绝用户请求，从而保障系统信息安全，适用于所有对外提供服务的智能体场景。

2.4.6 深度思考

部分大模型支持“深度思考”功能，开发者可通过开启 / 关闭该功能，灵活调控智能体的推理质量与 Token 消耗。该功能默认处于开启状态。例如，DeepSeek 在回答问题时，会先显示一段它的思考过程，比如“用户问的是 XX，我得先梳理 XX 要点，再结合 XX 信息，才能给出准确答案”，而不是直接给出最终结果。注意，当前仅部分模型支持深度思考功能。深度思考功能的核心逻辑与配置说明如下：

1 开启深度思考

开启深度思考后，智能体在响应用户请求时，将先进行一段不可见的思维链（Chain of

Thought）推理过程——逐步拆解问题、梳理推理逻辑、验证结论合理性，再生成最终回复。这样可显著提升复杂问题的解答准确性；但因额外增加了推理步骤，会消耗更多 Token，且响应速度略有下降。

开启深度思考后，需注意以下约束：

- 模型不支持工具调用能力。
- 智能体无法使用插件、触发器、变量、数据库、文件盒等扩展能力，也无法添加工作流与对话流。
- 不支持使用与插件调用、工作流触发等相关的快捷指令。

2 关闭深度思考

关闭深度思考后，智能体将跳过思维链推理过程，直接生成最终回复，可有效降低 Token 消耗，提升响应速度，但复杂问题的解答准确性可能下降；适用于简单事实查询、基础指令执行等对推理深度要求较低的场景。

3 自动模式

启用自动模式后，模型将根据对话内容的复杂度，智能判断是否启用深度思考：

- 简单问题（如“北京的简称是什么”“计算 1+1 的结果”等事实查询、基础指令）：自动关闭深度思考，以便快速响应。
- 复杂问题（如“制定一份 3 天的上海旅游攻略”“分析某产品的市场竞争格局”等逻辑推理、创意生成类问题）：自动开启深度思考，保障回复质量。

2.5 扣子智能体插件设置

插件是扣子智能体平台提供的标准化工具集，每个插件可封装一个或多个工具（对应具体 API 接口），核心作用是拓展智能体的能力边界，弥补大模型在实时数据获取、特定功能执行等场景中的局限性。目前，扣子智能体平台已集成资讯阅读、旅游出行、效率办公、多模态处理等类型的插件资源，开发者可直接复用现有插件或自定义开发插件，快速实现智能体功能扩展。例如，为智能体集成天气查询插件，赋予其实时获取与查询天气数据的能力。

2.5.1 添加插件

插件需在智能体编排页面完成集成配置，具体路径为“编排”面板→“技能”模块。平

台支持手动添加与自动添加两种模式，开发者可根据业务需求选择适配方式。

1 手动添加插件（精准适配场景）

手动添加插件模式适用于开发者明确知晓所需插件的场景，可精准选择目标插件完成集成。操作步骤如下：

步骤 01 进入智能体编排页面，定位“技能”模块中的“插件”功能区，单击右侧的“+”图标，打开插件添加弹窗。

步骤 02 在弹窗中，可通过“个人空间”“团队空间”“插件商店”三个渠道筛选插件：个人空间中的是开发者自定义开发的插件，团队空间中的是团队共享的插件，插件商店中的是平台官方及第三方提供的公共插件。

步骤 03 若未找到适配插件，可单击弹窗底部的“创建插件”按钮，跳转至插件开发页面自定义开发（具体开发流程参考平台提供的插件开发文档）。

步骤 04 选中目标插件后，展开插件详情，查看包含的工具列表，确认功能匹配后单击“添加”按钮，完成插件与智能体的绑定。

2 自动添加插件（高效适配场景）

自动添加插件模式依托大模型的语义理解能力，根据智能体的人设与回复逻辑自动匹配插件，适用于开发者不明确插件选型的场景。操作步骤如下：

步骤 01 在“插件”功能区单击“自动添加”图标，触发大模型分析流程。

步骤 02 大模型将基于当前智能体的角色定位、核心功能描述，从插件商店中筛选最优匹配的插件并自动添加至智能体。

步骤 03 自动添加完成后，必须在“预览与调试”区域进行功能验证，确认插件可正常调用且符合业务需求。若结果不匹配，可删除不适配插件，重新通过手动模式添加目标插件。

自动添加插件操作界面如图 2-23 所示。

提示 添加插件后，需在“人设与回复逻辑”面板中明确插件调用规则（如“获取天气数据时，自动调用 DayWeather 插件”），确保大模型可在正确场景触发插件调用。单击扣子智能体平台页面左侧导航栏中的“资源库”，可查看当前空间资源库中所有可用的插件工具，便于跨项目插件复用。

图 2-23 自动添加插件

2.5.2 插件参数配置

插件添加完成后，需进行参数配置，核心目标是设置参数默认值与可见性。合理的参数配置可有效避免因参数缺失而导致的插件调用失败，同时减少大模型对固定参数的无效判断，提升调用效率。其中，参数默认值用于填充用户未输入的必要信息，参数可见性用于控制大模型是否可读取并修改该参数。

在插件调用流程中，若用户未提供插件所需的必要参数（如天气查询插件的“地域”参数），且未配置默认值，那么大模型将因参数缺失而无法正常调用插件，导致回复失败。若为该参数设置默认值（如“杭州市”），那么即使用户未输入地域信息，大模型也会自动使用默认值调用插件，确保功能正常执行。具体对比效果如下：

1 未设置默认值的效果

当用户未输入地域信息时，插件因参数缺失而无法调用，大模型无法返回有效结果。具体参数配置界面如图 2-24 所示。

对应的大模型回复效果如图 2-25 所示。

图 2-24 未设置默认值的效果

图 2-25 未设置默认值时大模型回复效果

2 设置默认值的效果

为“地域”参数设置默认值“杭州市”后，即使用户未输入地域信息，插件也可正常调用并返回默认地域的天气数据。具体参数配置界面如图 2-26 所示。

图 2-26 设置默认值的效果

对应的大模型回复效果如图 2-27 所示。

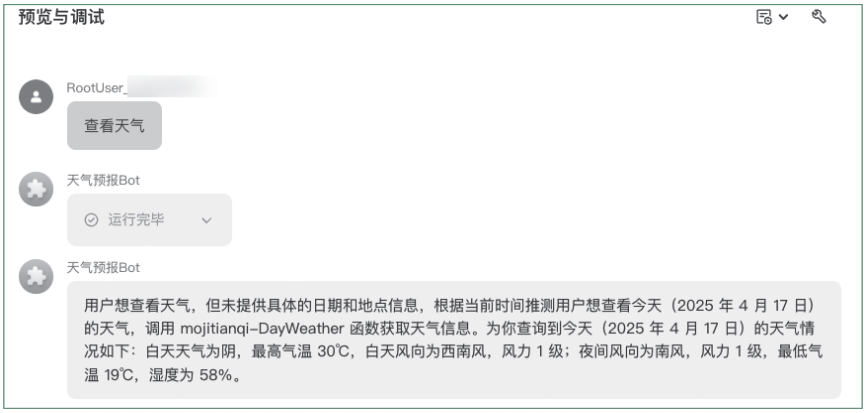


图 2-27 设置默认值时大模型回复效果

参数配置步骤如下：

步骤 01 在智能体编排页面的“插件”列表中，定位目标插件，单击插件右侧的“编辑参数”图标，如图 2-28 所示。

图 2-28 单击“编辑参数”图标

步骤 02 在打开的参数配置弹窗中，针对每个参数进行个性化设置。核心配置项说明如下：

- 默认值：支持设置固定值或引用变量值。固定值适用于参数取值稳定的场景（如地域默认值为“杭州市”）；变量值适用于需要动态获取的场景（如引用系统变量“当前城市”），变量相关说明可参考平台的“变量管理”文档。
- 可见性开关：开启开关时，参数对大模型可见，大模型可根据上下文动态修改参数值；关闭开关时，参数对大模型隐藏，大模型将强制使用设置的默认值，不可修改参数值。
- 配置规则补充：若设置默认值且开启可见性，则大模型将以默认值为基础，结合用户输入动态调整参数；若设置默认值且关闭可见性，则大模型仅能使用默认值，无法修改参数值。

参数配置完成后的界面如图 2-29 所示。

图 2-29 参数配置完成后的界面

2.5.3 绑定卡片数据

扣子可以为插件绑定固定的消息卡片模板，这样插件返回的内容就会按照这个固定模板展示（不再是单调的纯文本），数据会自动填充到卡片对应位置，看起来更清晰。绑定卡片数据的具体操作步骤如下：

步骤 01 触发卡片绑定功能。在目标插件右侧单击“绑定卡片数据”图标，如图 2-30 所示。

图 2-30 单击“绑定卡片数据”图标

步骤 02 选择卡片模板。在弹出的“智能体回复卡片配置”对话框中，可以选择两种卡片类型：一是“官方卡片”，即扣子官方提供的标准化卡片（如天气展示卡片、资讯列表卡片）；二是“我的卡片”，即通过“新增”自主设计的卡片。卡片模板选择界面如图 2-31 所示。

步骤 03 字段映射绑定。选中卡片模板后，将插件返回数据的字段（如天气插件的“城市名称”“温度”“风力”字段）与卡片元素（标题、正文、数据展示区）进行一一映射，完成后单击“保存”按钮即可生效。

图 2-31 卡片模板选择界面

2.5.4 删除插件

当插件不再满足业务需求时，可通过删除操作清理冗余资源，避免影响智能体配置管理。具体操作步骤：在智能体“插件”列表中定位目标插件，单击插件右侧的“移除”图标，系统将自动解除该插件与智能体的绑定，这样就完成删除操作。插件删除操作入口如图 2-32 所示。

图 2-32 插件删除操作入口

2.5.5 插件授权

部分插件（如飞书文档、企业微信相关插件）采用 OAuth 授权机制，简单来说就是得先获取开发者的账号权限，之后才能访问里面的资源。此类插件集成后，需完成授权配置，否则将导致插件调用中断。具体授权规则与操作流程如下：

- 开发阶段授权：在智能体或工作流中添加 OAuth 插件后，页面将显示未授权提示。单击提示后，按照页面引导完成开发者账号授权，即可正常进行调试与试运行。

- 发布后授权：绑定 OAuth 插件的智能体发布后，默认情况下每个用户在调用插件时需使用个人账号完成授权；若为工作流中的 OAuth 插件节点设置了“共享授权模式”，则用户在使用时将默认复用开发者账号的授权信息，无须手动授权。

OAuth 插件授权提示界面如图 2-33 所示。

图 2-33 OAuth 插件授权提示界面

2.6 小结

本章首先明确了扣子编程与扣子智能体平台的关系：扣子智能体平台是扣子编程平台的核心模块，聚焦智能体与工作流开发，提供一站式 AI Agent 搭建能力；其次，详细介绍了扣子智能体开发平台的核心体系与实践方法，从平台基础认知（空间、智能体等核心概念及关系）切入，通过天气查询机器人案例演示了智能体从创建、提示词编写、插件集成到调试发布的完整低代码搭建流程；后续又详细拆解了提示词设置（四种配置方式）与大模型配置（模型选型、协议选择等六大核心维度）的关键要点；最后深入讲解了插件生命周期管理（添加、参数配置、卡片绑定等）。

智能体的复杂任务执行往往需要依托工作流实现多步骤逻辑编排，因此下一章将聚焦扣子工作流，深入探讨工作流的搭建、配置与优化方法，进一步完善扣子智能体平台的开发能力体系。

CHAPTER

第 3 章

扣子工作流核心：Agent 的“任务流程图”

本章首先介绍扣子工作流的基本概念和特点，然后搭建一个简单的工作流，最后介绍工作节点的核心要素和配置方法，包括基础节点（开始和结束节点、大模型节点、插件节点）和业务逻辑节点（批处理节点、选择器节点和代码节点）等内容。

3.1 扣子工作流概述

扣子工作流（在扣子编程中也被称为低代码工作流）是一套结构化的可执行指令集合，用于承载业务逻辑，为 AI Agent 的数据流转、任务调度及逻辑执行提供标准化框架。低代码工作流的核心优势，在于将大语言模型的生成式能力与具象化业务逻辑深度耦合，通过系统化、流程化的节点编排，实现高效、可复用、可扩展的 AI 应用快速开发，在降低技术门槛的同时保障业务落地效率。

扣子智能体平台中提供了可视化拖曳画布，支持开发者通过节点拖曳快速搭建工作流，并可在画布内完成实时调试、节点联动校验，清晰呈现数据流转路径、任务执行时序及各节点的依赖关系，实现工作流的可视化开发与运维。

3.1.1 工作流类型

扣子平台提供两种核心工作流类型，分别适配不同业务场景：

- 工作流（Workflow）：聚焦功能类任务处理，通过节点的顺序执行、分支调度实现特定功能闭环，适用于数据自动化处理、批量任务执行等场景。例如，行业调研报告生成、海报自动化设计、绘本批量制作、数据清洗与分析等。
- 对话流（Chatflow）：基于对话交互场景的特殊工作流，专为对话类任务设计，通过多轮对话交互承载复杂业务逻辑，适用于 Chatbot、智能助手、智能客服、虚拟伴侣等对话式 AI 应用，可实现用户指令的分步拆解、上下文感知的响应生成。

3.1.2 节点：工作流的核心执行单元

工作流的核心载体是节点，每个节点均为具备独立功能的模块化组件，对应任务执行中的单个步骤或逻辑单元，负责数据接收、处理、运算及输出，实现工作流的原子化拆解与组装。简单地说，工作流就像一条完整的生产线，而节点就相当于生产线上的一个个工位，每个工位都有专属职责，只负责完成某一项具体工作。

所有工作流默认包含以下两个基础节点，用于保障工作流的完整生命周期：

- 开始节点：工作流的启动入口，用于定义工作流启动所需的输入参数、参数类型及校验规则，明确任务执行的初始条件。
- 结束节点：工作流的终止出口，用于接收各前置节点的处理结果，汇总输出工作流的最终执行结果，完成任务闭环。

节点间可通过输出引用实现无缝联动，形成完整的操作链路。例如，代码节点可在输入参数中直接引用大模型节点的输出结果，实现数据的跨节点传递与复用。这种联动关系可在可视化画布中直观呈现，便于开发者梳理逻辑。代码节点配置示例如图 3-1 所示。

图 3-1 代码节点配置示例

节点的灵活性与扩展性是提升工作流编排效率的关键。扣子工作流支持全类型节点（包括开始节点、结束节点、输出节点、插件节点、子工作流节点、代码节点、SQL 自定义节点、数据操作节点、问答节点、批处理节点、循环节点、变量聚合节点、变量节点、选择器节点等），适配多类数据（涵盖 String、Integer、Number、Boolean、Object、File、Array 等），无须开发者额外开发数据转换逻辑，可根据业务需求灵活选型，大幅提升了工作流编排的效率与适配性。

3.2 工作流与对话流的核心差异

工作流与对话流同属扣子工作流体系，二者基于统一的节点编排逻辑，但针对不同业务场景进行了差异化优化：工作流聚焦功能类任务的自动化执行，对话流聚焦对话类场景的交互逻辑落地。在应用开发中，通过集成对话流，可将用户对话指令拆分为标准化步骤节点，再结合自定义 UI 设计，可快速搭建适配移动端、网页端的对话式 AI 应用，实现对话流程的自动化、智能化。

相较于工作流，对话流的核心优势在于具备会话上下文管理能力。每个对话流均与一个独立会话绑定，在运行过程中可实时读取会话历史消息，并将本次执行结果同步记录至会话中，形成具备“记忆能力”的对话。工作流与对话流的具体差异如下：

- 若需搭建具备上下文感知与会话管理能力的智能体，可根据具体任务类型灵活选择工作流或对话流。
- 若需搭建对话式 AI 应用（如智能助手、智能客服等），推荐优先使用对话流，其内置的大模型节点支持会话历史读取、会话生命周期管理，同时可配置个性化对话 UI，支持快速发布至各类社交通信渠道。
- 若需搭建工具类 AI 应用（如数据批量处理、任务流程自动化等），推荐优先选择工作流，其节点编排更侧重任务的顺序执行与批量调度，适配高效自动化处理场景。

3.3 搭建一个简单的扣子工作流

结合前文所述，扣子工作流作为结构化的可执行指令集合，核心用于业务逻辑落地与特定任务。为让搭建流程更易理解，下面将以“生成行业资讯摘要 + 翻译英文摘要”的简单工作流为例（适配职场、新媒体等常见场景，不需要复杂代码，仅用到大模型节点 + 插件工具节点，新手可快速上手），详细拆解工作流的创建、编排、测试与发布全流程。完整工作流如图 3-2 所示。

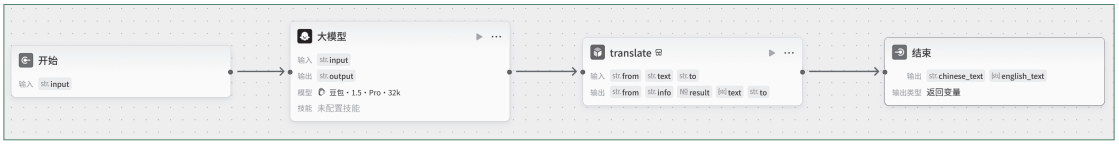


图 3-2 完整工作流

3.3.1 搭建流程详解

本次案例目标：搭建一个可输入中文行业资讯原文的工作流，该工作流先通过大模型节点生成简洁中文摘要（控制在 100 字内），再通过插件工具节点（翻译插件）将中文摘要翻译成英文，无须手动处理。

1 创建工作流

具体操作步骤如下：

步骤 01 登录扣子智能体平台（入口地址：https://www.coze.cn/space/）。

步骤 02 在页面顶部选择目标工作空间，在左侧导航栏中单击“资源库”进入资源管理页面。

步骤 03 在资源库页面右上角单击“+ 资源”按钮，选择“工作流”，进入“创建工作流”页面。

步骤 04 填写工作流名称与功能描述（名称：Industry_News_Trans；描述：输入中文行业资讯原文，自动生成 100 字内中文摘要，并将摘要翻译成英文，实现资讯处理自动化，提升办公效率），单击“确认”按钮完成创建，如图 3-3 所示。

图 3-3 创建工作流

注意 工作流名称需简洁明确，功能描述需精准覆盖核心能力，便于大语言模型理解工作流的应用场景与执行目标，这样调用起来也更准确。

创建完成后，系统将自动跳转至工作流编辑页面，初始状态下将默认生成开始节点与结束节点，构成工作流的基础骨架。结合本次案例，我们需为这两个基础节点补充核心配置：

开始节点负责接收“中文行业资讯原文”这一输入参数（供后续节点调用）；结束节点负责汇总“中文摘要”“英文翻译结果”，作为最终结果输出。具体配置如图 3-4 所示。

图 3-4 为两个基础节点补充核心配置

2 配置工作流

对于这个简单工作流来说，我们仅需用到两个核心节点——“大模型”节点（生成中文摘要）和“插件”节点（翻译插件，实现英文翻译），即可完成编排配置。具体操作如下：

步骤 01 添加并配置“大模型”节点：在工作流编辑页面底部单击“+ 添加节点”按钮，在弹出的节点选择面板中单击“大模型”，即可在画布中添加“大模型”节点（选用豆包系列大模型即可）；将其拖曳至“开始”节点与“结束”节点之间；单击该节点，在弹出的参数配置面板中新建大模型输入变量 {{input}}，并将该变量配置为开始节点的输入（路径：开始节点 -input）。大模型的提示词和相关配置如图 3-5 所示，我们还可以使用 AI 工具优化提示词。

步骤 02 添加并配置插件节点（翻译插件）：在工作流编辑页面底部单击“+ 添加节点”按钮，在弹出的节点选择面板中单击“插件”，搜索“文本翻译”插件，选择一个使用量较高的文本翻译插件，即可在画布中添加文本翻译节点 translate；将其拖曳至“大模型”节点与“结束”节点之间；将文本翻译节点 translate 的 {{text}} 变量配置为大模型的输出（路径：大模型 -output），{from} 和 {to} 变量根据实际情况配置即可，文本翻译节点的完整配置如图 3-6 所示。

图 3-5 大模型的提示词和相关配置

图 3-6 文本翻译节点的完整配置

步骤 03 参数校验：确认三个节点联动正常，“大模型”节点可接收原文、输出摘要，文本翻译节点 translate 可接收摘要、输出英文，“结束”节点可同时接收中文摘要与英文翻译结果，确保数据流转无误。

3 测试并发布工作流

工作流配置完成后，需通过测试验证摘要生成与翻译效果，确认无误后再发布。具体操作如下：

步骤 01 在工作流编辑页面底部单击“试运行”按钮，启动测试流程；本次案例的输入参数为文本类型，无须上传文件，直接输入测试数据。输入的行业资讯如下：

2026 年初，TCL 科技旗下华星光电全面落地其自研的“星智大模型 3.0”，这款专为半导体显示行业打造的垂直 AI 系统已深度嵌入面板研发与制造全流程，不仅能通过自然语言交互自动解析如色偏、亮度不均等复杂不良现象，并逆向定位到蒸镀、光刻或封装等具体制程环节，还能结合内嵌的数万份专利文献、材料数据库和失效分析报告进行多物理场联合仿真，在新型 OLED 发光材料开发中将原本需 6 个月的试错周期压缩至 4 个月内，同时在 G8.5 高世代产线上实现制程参数的实时动态优化，使整体良品率提升 1.2 个百分点，单线年增利润超 5 亿元，其 AI 智能体还能自主调用内部 EDA 工具、视觉检测系统和能耗管理平台，形成从问题发现、根因分析、方案生成到执行验证的闭环，2025 年全年累计减少废品损失约 3 亿元、节省材料打样成本超 1 亿元，目前该模型已支持 Mini-LED、Micro-OLED、印刷显示等全技术路线，并计划向供应链开放部分能力以构建产业级 AI 生态，被工信部列为人工智能赋能新型工业化的标杆案例，标志着国产 AI 正从“炫技式演示”转向“扎根产线、创造真金白银价值”的深水区。

步骤 02 在试运行过程中，系统将实时展示节点执行状态，执行成功的节点边框将显示绿色；在“大模型”节点右上角单击，可查看输入的资讯原文与生成的中文摘要，验证摘要是否符合 100 字内、核心信息完整的要求；在文本翻译节点 translate 的右上角单击，可查看输入的中文摘要与输出的英文翻译结果，验证翻译准确性；单击“结束”节点，可查看汇总后的中文摘要与英文翻译结果，确认整体流程无误。试运行的结果如图 3-7 所示。

图 3-7 试运行的结果

步骤 03 测试通过后，单击“发布”按钮。发布时，可选择将本次试运行阶段保存的测试集（资讯原文、中文摘要、英文翻译结果）作为默认测试集；发布完成后，同一工作空间内的其他用户可调用该工作流，输入其他中文行业资讯，快速生成对应的中文摘要与英文翻译，提升资讯处理效率。

3.3.2 在智能体中添加工作流

工作流发布后，可在目标智能体中添加并调用，实现智能体能力的扩展。具体操作步骤如下：

步骤 01 进入当前工作空间的智能体管理页面，选择需要添加工作流的目标智能体，进入智能体编排页面。

步骤 02 在智能体编排页面的工作流管理区域，单击右侧的“+”图标，打开“添加工作流”对话框，如图 3-8 所示。

图 3-8 打开“添加工作流”对话框

步骤 03 在该对话框中，选择已发布的自建工作流，单击“添加”按钮完成添加，如图 3-9 所示。

步骤 04 在智能体的“人设与回复逻辑”面板中，通过引用工作流名称实现工作流的触发与调用，完成智能体与工作流的联动，如图 3-10 所示。

图 3-9 添加工作流

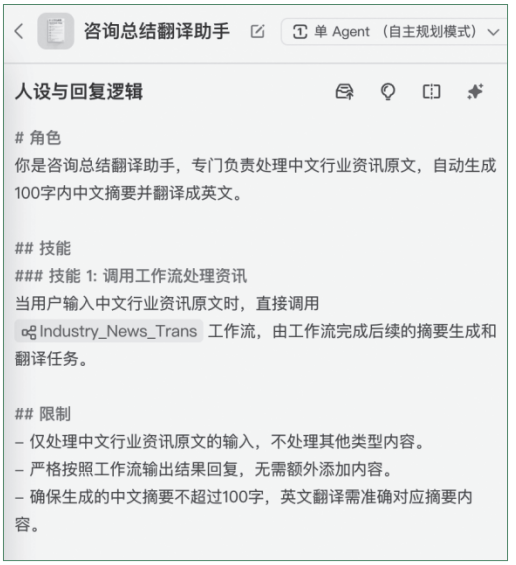


图 3-10 智能体与工作流联动

3.4 工作流节点

在前面的简单工作流例子中，我们用到的大模型、插件等，其实都是工作流的节点，而且是大部分工作流会用到的核心节点。本节将对主要工作流的核心节点进行详细介绍；对于非核心的节点，我们会在后续的案例讲解中进行详细介绍和实操演示。工作流的核心节点分为基础节点和业务逻辑节点两类。

3.4.1 基础节点

基础节点是扣子工作流搭建的基石，主要包括开始节点、结束节点、大模型节点和插件节点四类。其中，开始节点负责定义工作流启动所需的输入信息，结束节点用于返回工作流运行后的最终结果，大模型节点依托大语言模型能力处理文本生成等自然语言任务，插件节点则通过调用各类插件拓展工作流的功能边界。这四类节点协同支撑大多数基础工作流的正常运行。

1 开始节点

开始节点作为工作流的起始组件，主要用于定义启动工作流所需的输入信息。该节点仅

包含输入参数，不具备输出等其他类型参数。开始节点默认包含一个输入变量 {{input}}，用于接收用户在本轮对话中输入的原始数据，开发者可根据实际业务需求，额外添加其他输入变量。

开始节点配置说明如下：

- 变量类型：开始节点支持配置 String、Number、Object 等多种类型的输入变量，其中 Object 类型变量最多支持 3 层嵌套结构。
- 变量设置方式：支持直接新增变量并定义变量名称，也可通过导入 JSON 数据的方式批量添加输入变量。具体操作如下：单击“JSON 导入”图标，在弹出的面板中输入 JSON 数据，再单击“同步 JSON 到节点”按钮，即可完成输入变量的自动导入，如图 3-11 所示。

图 3-11 输入变量的自动导入

- 变量描述：变量的描述信息，帮助模型理解传入变量的含义。将工作流绑定到智能体中使用时，模型会自动分析用户的问题，并将问题中表达的信息填入对应的参数中。
- 是否必填：变量是否必填。如果未指定必填变量，将无法执行工作流。将工作流绑定到智能体中使用时，若用户问题中缺少必填变量，则不会触发工作流。

2 结束节点

结束节点为工作流的终止组件，用于反馈工作流运行后的最终结果。该节点支持两种返回模式，分别为返回变量和返回文本。

1）返回变量

在返回变量模式下，工作流运行结束后，将以 JSON 格式输出所有返回参数。该模式适用于工作流绑定卡片或作为子工作流的应用场景。若工作流直接与智能体绑定，当对话触发工作流运行时，大模型会自动对 JSON 格式的输出内容进行总结，并以自然语言形式反馈给用户。返回变量支持配置 String、Number、Object 等多种类型，其中 Object 类型最多支持 3 层嵌套结构。

2）返回文本

在返回文本模式下，工作流运行结束后，智能体中的模型将直接采用预设内容响应对话。回复内容支持引用输出参数，同时可配置流式输出功能，具体说明如表 3-1 所示。

表3-1 返回文本模式下的输出设置

设　置	说　明
输出变量	输出节点中的输出参数。为工作流绑定卡片时可以使用这些参数
回答内容	工作流的最终输出内容，不可设置为空。 • 支持引用输出参数，引用方式为 {{变量名}}。 • 支持开启流式输出：开启后，回复内容中由大模型生成的内容将逐字显示在对话中，类似打字机的效果。 流式输出适用于输出文本较长或需要工作流即时反馈的场景，呈现实时对话的交互效果，用户无须等待一大段文字一次性加载，可显著提高对话过程中的用户体验

3 大模型节点

大模型节点是扣子工作流的核心基础节点之一，核心功能为调用大语言模型完成各类自然语言处理任务，其中大部分配置和 2.4 节中的基本相同。

大模型节点依托大语言模型的自然语言理解与生成能力，可处理各类复杂的自然语言处理任务。开发人员可结合具体业务场景需求，选择适配的模型类型，并通过配置提示词定义模型的人设与回复风格。为提升模型生成结果的精准性，还可在节点中配置模型相关参数，以调控模型回复的文本长度、内容多样性等输出配置。

1）模型

大模型节点的输出内容质量很大程度上受模型能力的影响，建议根据实际业务场景选择模型，如图 3-12 所示。可选的模型范围取决于当前的账号类型：

- 个人版（个人免费版、个人进阶版、个人高阶版、个人旗舰版）用户仅可使用默认的几类模型，且存在对话数量限制。

- 企业版（企业旗舰版、企业标准版）用户在个人版基础上，还可以使用火山引擎方舟平台的模型。

2）技能

大模型节点支持配置技能扩展，可集成插件、工作流及知识库等组件，以此拓宽模型的功能边界，如图 3-13 所示。技能配置完成后，大模型节点的功能特性将趋近于独立运行的智能体，具备自动意图识别、技能调用时机判断及调用方式选择的能力，可显著提升节点的文本处理效率与文本生成质量，同时简化工作流的节点编排复杂度。以天气穿搭推荐场景为例，传统实现方式需先通过插件节点查询目标区域的天气数据，再由大模型节点结合天气信息生成穿搭建议，适合需明确拆分步骤、精准控制每个环节的场景；在技能配置模式下，可直接在大模型节点中集成天气查询插件，由节点自动完成天气查询与穿搭建议生成的全流程处理，适合追求操作简洁、无须手动干预的场景。

图 3-12 选择模型

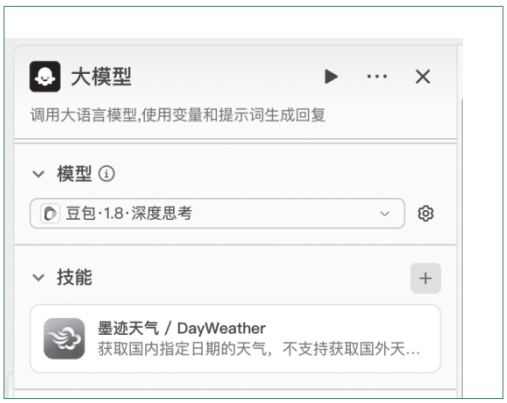


图 3-13 配置技能

3）输入

输入参数是需要嵌入提示词的动态内容，系统提示词与用户提示词均支持引用输入参数，以实现提示词的动态适配与调整。在配置输入参数时，需明确变量名称与变量值，其中变量值既可设为固定值，也可引用上游节点的输出参数，如图 3-14 所示。

4）视觉理解输入

针对支持视觉理解能力的模型，可配置图片类型的输入参数。图片类型数据有两种传入方式：一是直接上传图片文件；二是通过变量传入图片 URL，传入后可在提示词中对该类变量进行引用。例如，“图片 {{ 变量名 }} 中有什么？”视觉理解输入功能适用于图像识别、图像分析及文档处理等相关场景，可实现图像内容的解析与识别——提取图像中的物体、场景、文字等核心信息，并基于提取结果生成标准化描述或执行预设定任务。视觉理解输入参数如图 3-15 所示。

图 3-14 配置输入参数

图 3-15 视觉理解输入参数

5）系统提示词

系统提示词用于定义大模型的“基础人设”和“回复规则”，是模型执行所有任务的前置约束，相当于给模型设定的“固定工作准则”，如图 3-16 所示。其配置方式有三种：直接插入提示词库中的现成模板、插入资源库中已创建的提示词，或手动自行编写。编写规范可参考前文关于提示词的相关介绍。编写系统提示词时，可引用输入参数中的变量及已添加至大模型节点的技能（如插件、工作流、知识库），以提升提示词编写效率。具体引用格式示例：{{variable}} 用于直接引用变量，{{ 变量名 . 子变量名 }} 用于引用 JSON 格式的子变量，{{ 变量名 [数组索引]}} 用于引用数组中的指定元素。

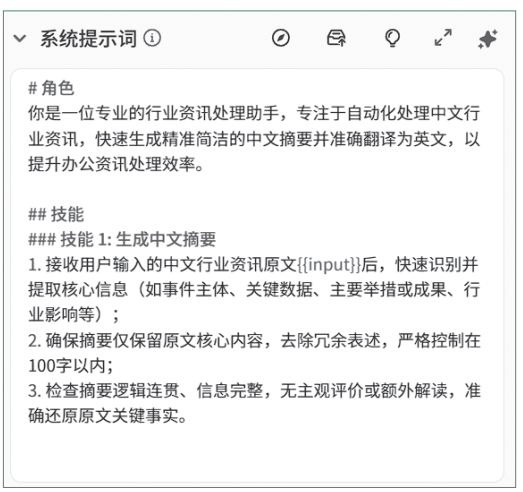


图 3-16 系统提示词

6）用户提示词

与系统提示词的“固定准则”不同，用户提示词是本轮对话中具体的“任务指令”，相当于开发者或用户下达给模型的“单次工作要求”，用于明确模型本次需要完成的具体任务、解答的具体问题。和系统提示词一致，用户提示词也可引用输入参数中的变量，适配动态任

务需求，如图 3-17 所示。二者的核心区别在于：系统提示词是“长期固定的人设规则”，全程约束模型；用户提示词是“单次具体的任务指令”，仅针对本次任务生效。

在后续多数实战案例中，用户提示词可直接填写嵌入的变量名，无须额外编写复杂指令。

7）输出

输出用于指定大模型节点的输出内容格式及输出参数配置，如图 3-18 所示。大模型节点支持以下三种输出格式：

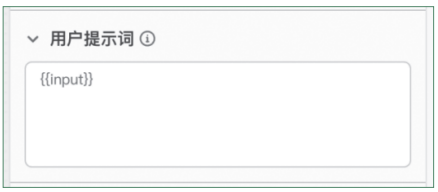


图 3-17 用户提示词

图 3-18 输出参数配置

- 文本格式：采用纯文本形式输出，此时大模型节点仅包含一个输出参数，该参数值为模型生成的纯文本响应内容。
- Markdown 格式：采用 Markdown 标记语言格式输出，此时大模型节点仅包含一个输出参数，该参数值为模型生成的 Markdown 格式响应内容。
- JSON 格式：采用标准 JSON 数据格式输出，支持两种参数结构配置方式：一是直接导入 JSON 样例，系统将根据样例格式自动解析并生成对应的输出参数结构；二是手动新增多个输出参数，并分别定义各参数的类型属性。

输出参数的名称定义与描述说明会直接影响模型输出结果与参数匹配的准确性。当需配置多个输出参数时，建议为每个参数设定具有明确业务含义的名称，并补充详细的描述信息，确保参数用途清晰可辨。

4 插件节点

插件是一系列实用工具的集合，而集合中的每一个工具本质上都是一个可直接调用的 API 接口（开发者无须理解 API 的复杂原理，只需知道它能实现特定功能即可）。开发者既可直接选用扣子插件商店中已上架的各类插件，也可使用自身资源库中的自定义插件。这些插件均支持以节点形式直接集成到低代码工作流中，无须额外配置复杂的关联关系。其核心作用就是拓展智能体与工作流的能力边界，让工作流跳出基础的逻辑处理和文本生成范畴，轻松实现查询、统计、推送、解析等各类具体业务操作，如查询天气、获取新闻、解析文档、发送消息等，从而大幅降低工作流的开发难度和周期。插件节点的大部分配置与 2.5 节中介绍的

基本一致，核心区别在于，插件节点作为工作流节点，额外增加了一些在工作流中的配置。

1）添加插件节点

插件节点的添加操作流程如下：在工作流编辑页面底部单击“+ 添加节点”按钮，在弹出的节点选择面板中选中“插件”，随后在弹出的“添加插件”页面中选择所需调用的插件，即可完成插件节点的初步添加。为提升配置效率，可通过收藏插件实现快速添加：在节点选择面板中定位至已收藏的插件，选中目标插件工具，即可快速完成插件节点的添加操作，如图 3-19 所示。

图 3-19　添加插件节点

2）配置插件节点

（1）输入与输出。

插件节点的输入与输出结构由插件本身定义，不支持开发者自定义配置。在配置插件节点时，必须为所有必选输入参数指定明确的数据来源，数据来源支持设置为固定值，或引用上游节点的输出参数。插件节点在运行过程中将自动调用对应插件工具处理输入参数，并严格按照插件工具定义的输出规范返回处理后的数据。开发人员可单击输出区域右上角的“查看示例”选项，查阅输出参数的详细说明及完整输出示例，明确参数含义与数据格式。例如之前用到的翻译插件，其输出示例如图 3-20 所示。

（2）模拟集。

在工作流试运行阶段，插件节点默认输出插件的真实运行数据；开发人员可选择启用模拟集功能，以模拟数据作为输出结果。模拟集是插件的模拟输出数据，启用后，工作流试运

行时无须调用插件，直接以模拟集数据作为下游节点的输入。模拟集支持开发人员自定义配置，也可通过 AI 自动生成。

图 3-20 翻译插件输出示例

（3）异常处理。

默认配置下，若插件节点出现运行超时、执行异常等情况，工作流将立即中断，相关错误信息将同步展示于工作流调试界面及 API 返回结果中。开发人员可手动配置节点异常处理规则，包括节点运行超时时间、异常重试次数及异常分支跳转逻辑等，如表 3-2 所示。

表3-2 异常处理配置

设 置 项	说 明
超时时间	超时时间指节点运行的最大耗时，如果超过此时长，则判断为节点运行超时。 • 默认值为180秒（3 分钟）。 • 可设置范围：0.1 ~ 600秒，用于灵活控制节点执行时限
重试次数	节点运行超时或异常时，默认不重试，也可以设置为重试 1 次
异常处理方式	节点运行超时或异常时，默认中断工作流。可手动修改为以下任意一种方式： • 中断流程：工作流立即终止，不再执行后续节点。 • 返回设定内容：发生异常后，工作流的运行不会中断。开发者可自定义需要返回的输出字段内容，必须是输出中已定义的字段，且为合法的JSON格式。另外，节点还会返回输出参数 isSuccess、errorBody，传递节点异常的详细信息。 • 执行异常流程：发生异常后，工作流的运行不会中断，转而执行异常流程分析，开发者需要为新增的异常分支配置处理流程。异常信息会通过节点的输出参数isSuccess、errorBody返回

异常处理配置如图 3-21 所示。

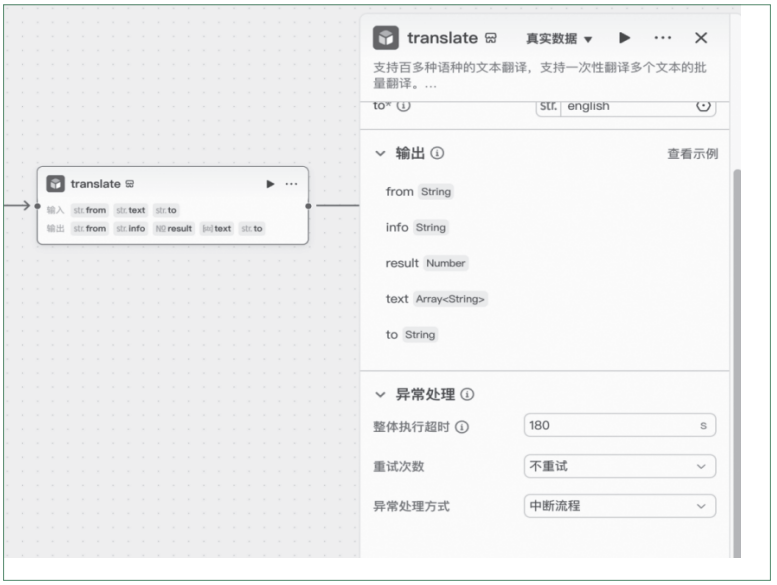


图 3-21 异常处理配置

3.4.2 业务逻辑节点

扣子工作流中的业务逻辑节点的核心作用是实现流程的复杂业务管控与数据批量 / 分支处理，支撑工作流跳出基础的线性执行模式，适配多场景、高复杂度的业务需求。本小节仅聚焦批处理节点、选择器节点、代码节点三类核心业务逻辑节点，系统介绍其功能定义、配置规范与应用场景；对于数据库节点、知识库节点等其他业务逻辑相关节点，将在后续实战案例章节中，结合具体业务场景进行详细介绍与实操演示，确保读者掌握各类节点的落地应用方法。

1 批处理节点

在工作流的执行过程中，各节点按预设顺序依次执行一次。针对需重复执行多次任务的场景（如批量查询多城市天气数据），可通过批处理节点提升工作流整体执行效率。批处理节点可批量调度除批处理节点、循环节点以外的所有节点，执行终止条件为达到预设次数限制或遍历完目标列表中的所有元素。批处理节点主要适用于大量数据的并行处理场景。相较于重复添加多个相同节点执行同类任务，批处理节点可显著提升任务执行效率与工作流编排

简洁度。以故事绘本批量生成场景为例，当需要基于每个故事片段生成对应配图时，可通过批处理节点调度文生图工作流，实现多幅配图的一次性批量生成。

注意事项：

- 批处理任务默认采用并行执行模式，若需实现串行执行，可将并行运行数量设置为 1，或选用循环节点替代实现。
- 批处理节点内部不支持嵌套配置另一个批处理节点或循环节点，避免出现流程调度冲突。
- 在低代码工作流中，同一时刻仅支持运行一个流式插件或消息节点，禁止通过批处理节点对该类节点进行批量并行调度，否则将导致节点输出异常，无法按预设逻辑返回结果。

1）配置批处理节点

批处理节点的处理对象为输入参数所引用的数组，典型示例为上游节点输出的多条数据结果。该节点将对数组中的每个元素进行遍历，并行执行预设处理逻辑，直至所有元素处理完毕，或达到预设的次数上限。

（1）批处理设置。

为规避并行运行数量过高引发的性能风险，批处理节点采用分批执行机制，默认每批执行 10 次，累计最大执行次数为 200。开发人员可通过批处理设置模块，配置每批最大执行次数及累计执行次数上限。具体参数说明如下：

- 并行运行数量上限：用于定义单批任务的最大执行次数，默认值为 10。该参数可配置为固定数值（如 5），也可引用上游节点输出的数值类型参数。注意，若引用的参数值大于 10，则系统自动校准为 10；若引用的参数值小于 1，则系统自动校准为 1。
- 批处理次数上限：用于定义批处理节点的累计最大执行次数，当累计执行次数达到该上限时，节点将立即终止运行。该参数默认值为 100，最大可配置值为 200。其执行逻辑为：节点优先对数组元素进行遍历，若累计执行次数达到预设上限，即使数组内仍有未处理元素，节点也将终止执行流程。

示例说明 若批处理节点的处理对象为长度 50 的数组，并行运行数量上限配置为 10，批处理次数上限配置为 30，则该节点将分 3 批执行，每批并行处理 10 个任务，累计执行 30 次后终止运行。

（2）输入。

批处理节点的输入参数的核心作用是为批处理体内部节点提供可逐一遍历的数据源，建

立批处理节点与批处理体之间的数据传递通道。批处理节点的核心处理对象为数组结构数据，其输入参数配置直接决定批处理体能否实现对数组元素的精准逐一遍历。输入参数存在严格约束：批处理节点至少需配置 1 个输入参数，且所有输入参数的数据源仅支持引用上游节点输出的 Array（数组）类型数据，不支持引用 String、Number、Object 等非数组类型数据，也不支持设置固定值作为输入参数。

假设上游节点输出一个存储 5 个城市名称的数组（[" 北京 "," 上海 "," 广州 "," 深圳 "," 杭州 "]），需要通过批处理节点批量查询这 5 个城市的天气。此时，需在批处理节点中定义输入参数（如命名为 cityList），并引用该上游节点的输出数组；配置完成后，批处理体内部的天气查询插件节点即可引用 cityList 中的单个城市元素，逐一遍历查询每个城市的天气。若未定义该输入参数，天气查询插件节点将无法获取单个城市名称，仅能获取整个城市数组，进而无法完成批量查询操作。

批处理体是批处理节点的核心执行载体，本质为独立的子流程画布，与循环体的执行逻辑具有同质性。每个批处理节点实例均会自动生成唯一对应的批处理体画布，用于承载批量处理任务的具体执行逻辑。批处理体的核心功能是容纳待批量调度的节点，并定义节点间的执行顺序，实现对输入数组元素的逐一遍历处理。例如，在批处理体内添加天气查询插件节点，配置该节点引用批处理节点的输入参数 cityList 中的单个城市元素，完成单城市天气查询逻辑；随后通过连接线明确批处理体与该插件节点的执行关系，同时完成批处理节点与上游城市数组输出节点、下游结果接收节点的连线配置，即可实现多城市天气的批量查询。批处理体将按预设逻辑逐一遍历 cityList 数组元素，驱动插件节点完成单次查询任务。

（3）输出。

批处理节点的输出参数会自动聚合批处理体内部节点的多次执行结果，生成标准化输出数据，为下游节点提供统一的批量处理结果数据源。例如，批处理节点输入参数 cityList 为包含 5 个城市名称的数组，批处理体内部的天气查询插件节点对每个城市逐一遍历查询，每次查询后都输出该城市的天气数据（如温度、湿度、天气状况）；批处理节点自动将这 5 次查询结果整合为数组格式并统一输出；下游节点可直接对该数组进行遍历、解析等后续处理，无须额外配置聚合逻辑。

2）批处理节点示例

结合前文批处理节点的输入、批处理体、输出的核心配置规范，下面介绍一个简单示例——通过视觉理解模型批量分析图片内容。该示例可直观体现批处理节点对数组（图片数组）元素的逐一遍历处理能力。其工作流整体编排流程如图 3-22 所示。

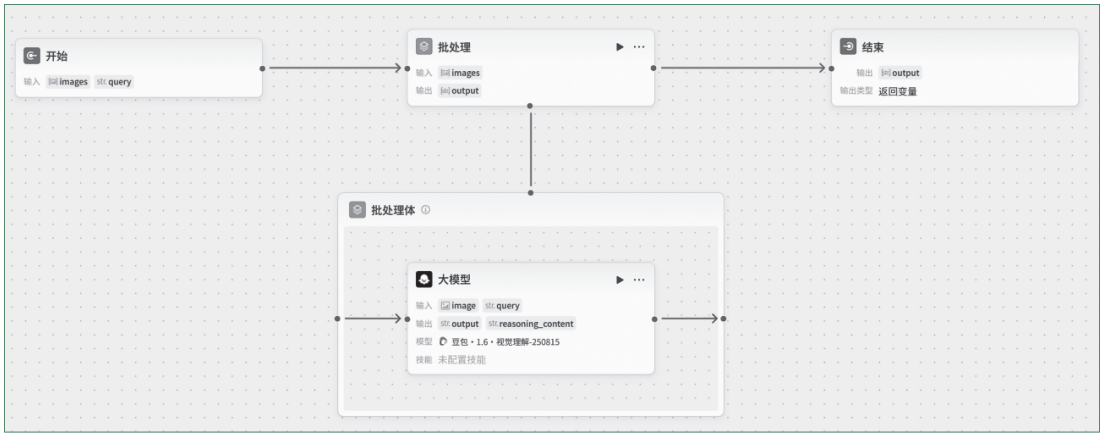
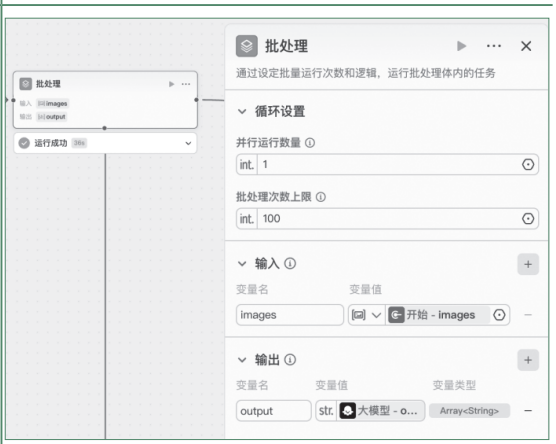


图 3-22 工作流整体编排流程

该示例工作流核心节点说明如表 3-3 所示。

表3-3 示例工作流核心节点说明

节点类型	说　明	示　例
开始节点	定义两个输入参数： • images：图片数组格式，支持上传多张图片。 • query：String格式 ，后续节点将作为模型的提示词	
批处理节点	设置输入参数image，引用开始节点的image参数，用于将图片数组传递给批处理体，实现对每张图片的循环处理	

（续表）

节点类型	说　明	示　例
批处理体	添加一个大模型节点，配置如下： • 模型：选择豆包视觉理解模型（支持图片内容分析）。 • 输入参数： - image：引用批处理节点中当前遍历的图片元素（如 item in input）。 - query：引用开始节点的query参数。 • 用户提示词：直接拼接两个参数，写作 {{query}}{{image}}	

2 选择器节点

选择器节点是扣子工作流中用于构建分支执行逻辑的核心业务逻辑节点，本质为条件判断节点（对应 if-else 逻辑），核心作用是基于预设条件判断，实现工作流执行路径的动态分支跳转，支撑工作流适配多场景差异化执行需求。

1）条件分支

选择器节点接收上游节点传入的参数后，将自动对预设的“如果”分支条件进行校验，若参数满足该分支条件，则驱动工作流执行“如果”分支对应的节点逻辑；若参数不满足该分支条件，则执行“否则”分支对应的节点逻辑。每个分支条件支持配置多个子判断条件，子判断条件之间可通过“且”“或”逻辑关系组合，满足复杂场景下的多条件联合判断需求。同时，选择器节点支持添加多个“如果”分支条件，开发人员可通过拖曳分支条件配置面板的排序，灵活设定各分支条件的校验优先级。

例如，选择器节点基于知识库查询结果实现差异化分支处理，其核心逻辑如下：上游知识库查询节点输出参数 outputList（用于存储查询结果），当知识库未查到匹配内容时，该参数值为 empty；选择器节点承接该参数后，通过条件判断实现分支跳转——若 outputList 为 empty（未查到内容），则关联对应节点，执行无结果兜底逻辑；若 outputList 不为 empty（查到内容），则关联其他节点，执行响应逻辑，如图 3-23 所示。

图 3-23 选择器节点基于知识库查询结果实现差异化分支处理

2）优先级

当选择器节点配置多个“如果”分支条件时，系统将按照预设的优先级顺序，对各分支条件逐个进行校验，一旦检测到某一分支条件被满足，将立即执行该分支对应的逻辑，不再校验后续分支条件；若所有“如果”分支条件均不被满足，则系统将自动执行“否则”分支对应的逻辑，如图 3-24 所示。

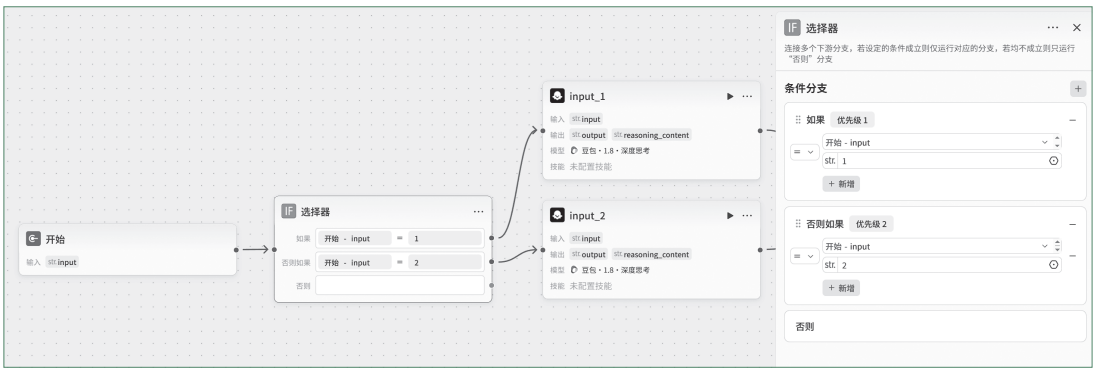


图 3-24 按照预设的优先级逐个检验各分支条件

3 代码节点

代码节点是扣子工作流中实现自定义逻辑的核心业务逻辑节点，支持开发者使用 JavaScript、Python 自定义代码片段。该节点接收上游节点传入的输入参数，经自定义逻辑处理后输出标准化结果，为工作流提供灵活的定制化处理能力。该节点适用于基础节点与插件节点无法覆盖的个性化数据处理场景，例如复杂数据格式转换、多源参数联动计算、自定义条件判断、特殊数据校验及处理等，可有效弥补标准化节点的功能局限性，支撑高定制化、高复杂度的工作流场景落地。

假设工作流上游节点输出参数 userInfo（JSON 格式，包含 name 和 age 字段，如 {"name":"张三 ","age":25}），需要通过代码节点提取该参数中的 name 字段，并拼接“你好，XXX”的问候语。该需求属于简单数据提取与字符串拼接，基础节点无法直接实现，可通过代码节点快速完成。代码节点的具体配置如图 3-25 所示。

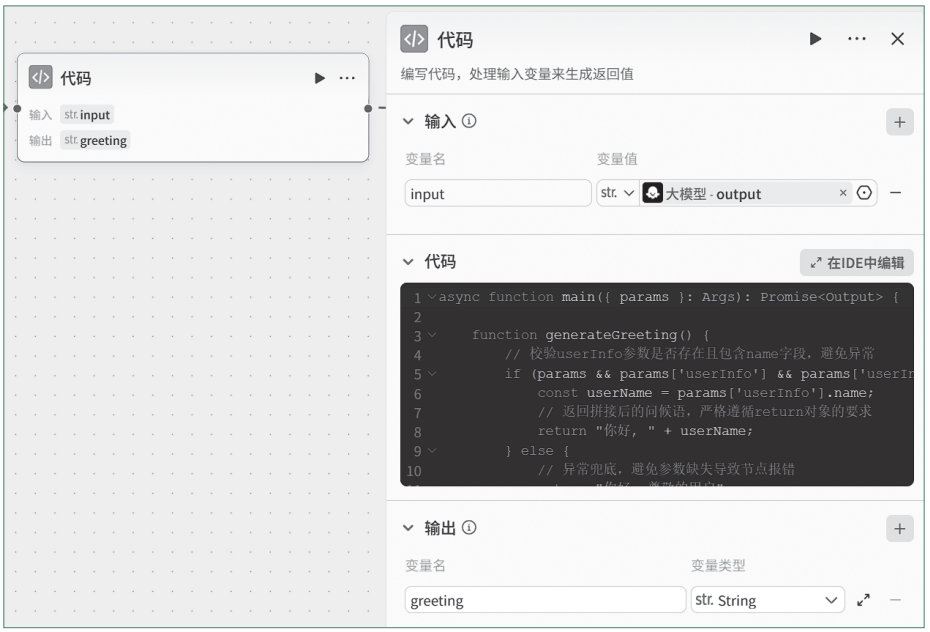


图 3-25 代码节点的具体配置

1）详细配置说明

代码节点的详细配置说明如表 3-4 所示。

表3-4 代码节点的详细配置说明

配　置	说　明
输入	声明代码中需要使用的变量。 • 添加输入参数时需设置变量名和变量值。 • 变量值支持： - 固定值（如字符串、数字） - 引用上游节点的输出参数（如 {{node1.output}}） • 在代码中通过 params['参数名'] 访问输入值，例如：params['input']

（续表）

配　置	说　明
代码	代码节点中需要执行的代码片段。 • 支持 JavaScript 和Python 两种语言。 • 引用变量：直接使用 params 中的输入参数。 • 返回要求： - 必须通过 return 一个对象来输出结果。 - 即使只有一个输出值，也必须包装为对象（如 return { result: value }）。 - 不支持定义多个函数（仅允许单个逻辑块或一个主函数）
输出	代码运行成功后输出的参数定义。 • 应根据实际需求，在输出结构中保留必要的参数。 • 必须确保：输出参数的名称和类型与代码中return的对象完全一致。 • 当节点的异常处理方式设为“返回设定内容”或“执行异常流程”时，系统会自动附加以下两个参数： - isSuccess（布尔值） - errorBody（异常详情） • 示例：若代码返回 { processedData: "abc" }，则输出配置中必须包含processedData字段

2）异常设置

代码节点异常设置用于管控节点运行过程中的超时、执行异常等场景，保障工作流运行稳定性。默认配置下，若代码节点出现运行超时、执行异常等情况，工作流将立即中断，相关错误信息将同步展示在工作流调试界面及 API 返回结果中。开发人员可根据业务需求，手动配置异常处理规则，具体包括超时时间、异常重试次数及异常分支跳转逻辑等，详细配置说明如表 3-5 所示。

表3-5 代码节点异常设置

设　置	说　明
超时时间	超时时间指节点运行的最大耗时，超过此时长将判定为节点运行超时。 • 默认值为 60 秒（1 分钟）。 • 可设置范围：0.1 ~ 60 秒，用于灵活控制节点执行时限
重试次数	节点运行超时或异常时，默认不重试，也可以设置为重试 1 次

（续表）

设　置	说　明
异常处理方式	节点运行超时或异常时，默认中断工作流。可手动修改为以下任意一种方式： • 中断流程：工作流立即终止，不再执行后续节点。 • 返回设定内容：工作流继续执行；开发者需提供合法的JSON格式的返回内容，且字段必须已在输出中定义。系统自动附加 isSuccess（false）和errorBody（异常详情）字段。 • 执行异常流程：发生异常后，工作流运行不会中断，转而执行异常流程分析，开发者需要为新增的异常分支配置处理流程。异常信息会通过节点的输出参数 isSuccess、errorBody 返回

代码节点的异常处理设置如图 3-26 所示。

图 3-26 代码节点的异常处理设置

3.5 小结

本章介绍了扣子工作流的定义、核心优势及搭建方法。工作流的核心节点分为基础节点与业务逻辑节点两大类，支撑各类工作流落地：基础节点包含开始、结束、大模型、插件四类，它们是工作流的基础并提供核心基础功能；业务逻辑节点重点介绍了批处理、选择器、代码三类，用于实现复杂流程管控与个性化处理，数据库、知识库等其他节点将在后续实战案例中详细介绍。