Agent 沙箱服务

MCP 使用指南

腾讯云

【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

MCP 使用指南
代码解释器 Code-MCP Server

最近更新时间：2025-09-28 18:50:42

Code-MCP Server 为 AI 代码助手提供安全的代码执行环境和沙箱管理工具。该服务器基于 AgentSandbox 沙箱技术，为大语言模型（LLM）提供安全隔离的代码运行环境，使 AI 代码助手能够安全地执行代码、管理文件和运行命令，从而简化开发工作流程。
将 Code-MCP Server 集成到 AI 代码助手中可以增强整个开发生命周期的工作流程，从简化初始代码测试和验证，到提供实时代码执行反馈。此外，它还通过安全的沙箱环境简化了复杂的代码调试和实验过程。所有这些都通过 AI 代码助手中的自然语言交互来简化复杂操作。

核心特性

- **安全代码执行**：在隔离的沙箱环境中执行多种编程语言的代码，包括 Python、JavaScript、TypeScript、R、Java。
- **文件系统管理**：提供完整的文件读写功能，支持在沙箱环境中创建、修改和读取文件。
- **命令行执行**：支持在沙箱环境中执行 Shell 命令和脚本。
- **会话管理**：基于会话的沙箱生命周期管理，支持长时间的交互式开发。

前置条件

- 获取 AgentSandbox API 密钥，您的密钥仅会在第一次生成时展示，我们不会存储您的密钥，请您妥善保存您的密钥。
- 创建名称为 code-interpreter-v1 的沙箱工具。

设置

AgentSandbox API 密钥获取

要使用 Code-MCP Server，您需要获取 API 密钥：

1. 注册腾讯云账号，并完成 实名认证 。
2. 在 Agent 沙箱服务 控制台中创建新的 API 密钥。
3. 记录您的 API 密钥，用于后续配置。

code-interpreter-v1 沙箱工具创建

1. 在 Agent沙箱服务 控制台中创建新的沙箱工具。
2. 在沙箱工具页面中，选择工具类型为“代码解释器”，并将工具名称设置为“code-interpreter-v1”，单击**确定**创建。

快速开始

本快速开始指南将引导您完成配置 Code-MCP Server 的步骤，以便与 Cursor IDE 等 MCP Client 进行集成。通过这些步骤，您将设置开发环境以利用 Code-MCP Server 的工具来管理安全的代码执行环境。

设置 Cursor

以下 JSON 为 Cursor IDE 的配置示例，您可以将如下配置写入 Cursor IDE 的 `mcp.json` 配置文件中。

> ⚠ **注意：**
> 请将下列的配置中的 `your-region` 替换为您的地域（如 ap-guangzhou），`your-api-key` 替换为您的 API 密钥。

```
{
  "mcpServers": {
    "agentsandbox-code-mcp": {
      "url": "https://mcp.your-region.ags.tencentcs.com/code?
api_key=your-api-key"
    }
  }
}
```

工具

Code-MCP Server 提供以下工具来管理安全的代码执行环境和文件系统操作。每个工具执行特定的操作，可以被调用来自动化沙箱环境中的常见任务。

代码执行

run_code

在安全沙箱中运行代码，使用 Jupyter 语法，在各语言的默认上下文中执行。

功能：

- 支持多种编程语言：Python、JavaScript、TypeScript、R、Java、Bash。
- 在隔离的沙箱环境中安全执行。
- 返回标准输出和错误输出。
- 支持交互式代码执行和状态保持。

参数：

- `code`（必需）：要执行的代码字符串。
- `language`（可选）：编程语言，默认为 "python"。

支持的语言：

- `python`：Python 代码执行。
- `javascript`：JavaScript 代码执行。
- `typescript`：TypeScript 代码执行。
- `r`：R 语言代码执行。
- `java`：Java 代码执行。
- `bash`：Bash 脚本执行。

返回值:

```
{
  "stdout": "标准输出内容",
  "stderr": "错误输出内容"
}
```

命令执行

run_command

在沙箱环境中执行 Shell 命令。

功能:

- 在安全的沙箱环境中执行任意系统命令。
- 支持复杂的命令行操作和脚本。
- 返回命令执行的输出和错误信息。

参数:

`command`（必需）：要执行的命令字符串。

返回值:

```
{
  "stdout": "命令标准输出",
  "stderr": "命令错误输出"
}
```

文件系统管理

write_file

向沙箱环境写入文件。

功能:

- 在沙箱文件系统中创建或覆盖文件。
- 支持任意文本内容写入。

- 自动创建必要的目录结构。

参数:

- `file_path`（必需）：文件路径。
- `content`（必需）：文件内容。

返回值:

无返回值。

read_file

从沙箱环境读取文件。

功能:

- 读取沙箱文件系统中的文件内容。
- 支持文本文件的完整内容读取。
- 返回文件的原始内容。

参数:

`file_path`（必需）：要读取的文件路径。

返回值:

文件的文本内容。