RELATÓRIO 🚀 \*\*Simplificando a Construção de Agentes AI com Google ADK: Um Guia Prático\*\*

═══════════════════════════════════════════════════════

Versão 1.0

Assunto Desenvolvimento, execução e depuração de \*\*Agentes de IA multi-agentes\*\* utilizando o \*\*Agent Development Kit (ADK) do Google\*\*, com foco em sua flexibilidade, tipologias (LLM, Workflow, Custom) e métodos de implantação.

Revisado por Apresentado por especialistas do \*\*Google ADK\*\*.

Data

═══════════════════════════════════════════════════════

RESUMO EXECUTIVO

═══════════════════════════════════════════════════════

Este guia prático apresenta o \*\*Google Agent Development Kit (ADK)\*\*, uma \*estrutura de código aberto\* essencial para a construção de agentes de IA. Demonstra como o ADK simplifica a criação, execução, avaliação e implantação de agentes complexos. Exploramos a arquitetura de um \*\*agente YouTube Short\*\* com seus sub-agentes (roteirista, visualizador, formatador), os diferentes tipos de agentes (LLM, Workflow, Custom) e os variados métodos de execução (CLI, Web UI, API e programático). Detalha-se a passagem de estado entre sub-agentes, a relevância dos \*agentes de fluxo de trabalho\* (Loop Agent) para um controle determinístico e as funcionalidades de \*serviços\*, \*runner\* e \*eventos\* para a execução programática. Uma jornada completa para desenvolvedores que buscam agilidade e robustez na construção de soluções de IA.

Palavras-chave: ['Google ADK', 'Agent Development Kit', 'Agentes AI', 'Multi-agentes', 'LLM Agent', 'Workflow Agent', 'Custom Agent', 'Python', 'Gemini 2.5 Pro', 'Debugging UI', 'Execução Programática', 'Serviços de Agente', 'Runner', 'Eventos de Agente', 'Desenvolvimento de Software de IA']

═══════════════════════════════════════════════════════

SUMÁRIO

═══════════════════════════════════════════════════════

1 Introdução 📋 3

2 Objetivo 🎯 3

3 Pré-requisitos e Acessos 🔑 4

4 Passo a Passo ⚙️ 4

5 Validação e Pontos de Controle ✅ 5

6 Solução de Problemas ⚠️ 5

7 Perguntas Frequentes (FAQ) ❓ 6

8 Considerações Finais 📄 7

═══════════════════════════════════════════════════════

1 INTRODUÇÃO 📋

═══════════════════════════════════════════════════════

A construção de sistemas com múltiplos agentes de IA pode ser uma tarefa desafiadora devido à sua inerente complexidade. Este guia introduz o \*\*Google Agent Development Kit (ADK)\*\* 🛠️, um \*framework de código aberto\* projetado para \*\*simplificar drasticamente\*\* esse processo. Em poucos minutos, você aprenderá não apenas a construir e executar seus próprios agentes de IA, mas também a \*\*configurar uma interface de usuário (UI) cliente\*\* para depurá-los com facilidade. O ADK destaca-se pela sua filosofia de \*\*'code-first flexibility'\*\*, oferecendo um controle granular incomparável através de construções de linguagem de programação e seguindo as melhores práticas de Python (classes e funções) para tornar o desenvolvimento de agentes tão intuitivo quanto o desenvolvimento de software convencional. Prepare-se para desmistificar a IA multi-agente!

═══════════════════════════════════════════════════════

2 OBJETIVO 🎯

═══════════════════════════════════════════════════════

O objetivo principal é fornecer o conhecimento e as ferramentas necessárias para:  
- Compreender o \*\*Google Agent Development Kit (ADK)\*\* e sua abordagem \*\*'code-first'\*\* na construção de agentes.  
- \*\*Construir e arquitetar agentes de IA\*\* complexos, utilizando sub-agentes e ferramentas integradas (como a ferramenta Google Search).  
- Explorar os \*\*diferentes tipos de agentes\*\* disponíveis no ADK: \*\*LLM Agent\*\*, \*\*Workflow Agents\*\* (Sequential, Parallel, Loop) e \*\*Custom Agents\*\*.  
- Dominar os \*\*quatro métodos de execução e interação\*\* com seus agentes: CLI (`ADK run`), UI Web (`ADK web`), API (`ADK API server`) e programaticamente.  
- Entender conceitos essenciais como \*\*passagem de estado\*\* entre sub-agentes (`output key`), \*\*serviços de sessão/memória/artefatos\*\*, o \*\*runner\*\* e os \*\*eventos\*\* para uma execução e depuração eficientes.  
- Depurar e otimizar o comportamento de agentes multi-agentes, assegurando que todos os sub-agentes sejam chamados conforme o esperado.

═══════════════════════════════════════════════════════

3 PRÉ-REQUISITOS E ACESSOS 🔑

═══════════════════════════════════════════════════════

LEGENDA DE RECURSOS NECESSÁRIOS

┌─────────────────────────────────────────────────────┐

│ RECURSO │ CAMINHO / DESCRIÇÃO │

├─────────────────────────────────────────────────────┤

│ 🔧 Power BI Desktop │ [Link On-line / Diretório] │

│ 📂 Pastas │ [\\caminho\principal\pasta] │

│ 📊 Excel │ [Excel\_Exemplo.xlsx] │

│ 📊 Word │ [Word\_Exemplo.docx] │

│ 🔐 SharePoint │ [Link SharePoint] │

│ 🌐 SAP │ [Nome da variante] │

└─────────────────────────────────────────────────────┘

═══════════════════════════════════════════════════════

4 PASSO A PASSO ⚙️

═══════════════════════════════════════════════════════

1. \*\*Introdução ao Google ADK e sua Filosofia\*\* 💡: Compreenda como o ADK, um kit de desenvolvimento de agentes de IA de código aberto, simplifica a construção, execução, avaliação e implantação de agentes. Ele destaca a flexibilidade 'code-first', que oferece controle granular sobre a orquestração de agentes por meio das melhores práticas de Python.  
2. \*\*Definição da Arquitetura do Agente Raiz e Sub-agentes\*\* 🧩: Projete e construa um 'Agente YouTube Short' composto por três sub-agentes:  
 \* \*\*Script Writer Agent\*\*: Responsável por gerar roteiros a partir de uma ideia inicial, utilizando a ferramenta Google Search integrada para pesquisar tendências.  
 \* \*\*Visualizer Agent\*\*: Encarregado de criar descrições visuais detalhadas com base no roteiro fornecido.  
 \* \*\*Formatter Agent\*\*: Combina o roteiro e as descrições visuais em um formato Markdown elegante e estruturado.  
3. \*\*Configuração do Ambiente de Desenvolvimento\*\* 💻: Instale o pacote `google-adk` utilizando `pip`. Crie o arquivo `agent.py` para definir o agente raiz (YouTube Short Agent) e seus sub-agentes, especificando o modelo LLM (ex: Gemini 2.5 Pro), descrição, instruções detalhadas e ferramentas a serem utilizadas.  
4. \*\*Gerenciamento de Estado entre Sub-agentes\*\* 🔗: Explore e utilize o conceito de `output\_key` para facilitar a passagem de informações e o armazenamento de respostas de LLMs (ex: `generated\_script`) entre os sub-agentes, permitindo que cada um acesse os dados gerados pelos anteriores. As instruções dos agentes podem ser carregadas de arquivos externos para uma organização mais eficiente do código.  
5. \*\*Autenticação do Modelo LLM\*\* 🔑: Configure as variáveis de ambiente necessárias (como uma chave de API) para autenticar o modelo LLM selecionado (ex: Gemini 2.5 Pro), garantindo o acesso seguro e autorizado.  
6. \*\*Execução Inicial do Agente via CLI (`ADK run`)\*\* 🏃: Execute o agente diretamente pela linha de comando (`ADK run `) para observar e analisar a resposta inicial gerada.7. \*\*Interação com o Agente via Web UI (`ADK web`)\*\* 🌐: Inicie a interface de usuário web (`ADK web`) para interagir visualmente com o agente, monitorar eventos e rastrear transferências de controle entre agentes. \*\*Identificação da Limitação:\*\* Observe que, inicialmente, apenas o sub-agente 'Script Writer' responde, pois o agente pai (do tipo LLM Agent) toma a decisão de que este é suficiente para a tarefa.8. \*\*Compreensão dos Tipos de Agentes no ADK\*\* 📊: Estude os diferentes tipos de agentes oferecidos pelo ADK: \* \*\*LLM Agent\*\*: O agente padrão que utiliza um LLM e ferramentas, empregando raciocínio para selecionar e orquestrar sub-agentes. \* \*\*Workflow Agents\*\*: Projetados para controle determinístico, permitindo definir a sequência exata de execução dos sub-agentes (Sequential, Parallel, Loop). \* \*\*Custom Agents\*\*: Oferecem flexibilidade máxima, possibilitando a combinação de diversos elementos para construir agentes altamente personalizados e complexos.9. \*\*Refatoração para Workflow Agent (Loop Agent)\*\* 🔄: Substitua o agente pai do tipo LLM Agent por um `Loop Agent` para assegurar que todos os sub-agentes sejam executados de forma iterativa. Remova parâmetros como modelo, descrição e instruções, que são desnecessários para um Workflow Agent, e adicione o parâmetro `maximum\_iterations` para controlar o número de iterações.10. \*\*Verificação da Execução do Loop Agent via Web UI\*\* ✅: Execute o agente refatorado com o `Loop Agent` utilizando `ADK web` e confirme que todos os sub-agentes (Script Writer, Visualizer, Formatter) são invocados iterativamente, inspecionando a lista de eventos exibida na UI.11. \*\*Execução Programática do Agente\*\* 👩‍💻: Aprenda a invocar o agente de forma programática: \* Entenda os \*\*Serviços\*\* (memória, sessão e artefatos, essenciais para a persistência de conversas e outputs). \* Compreenda o \*\*Runner\*\* (o 'coração' da execução do agente, responsável por iniciar e gerenciar o fluxo). \* Conheça os \*\*Eventos\*\* (ações atômicas transmitidas assincronamente pelo runner, permitindo monitoramento detalhado). \* Instancie um serviço de sessão em memória (`InMemorySessionServices`), configurando `app\_name`, `user\_id` e `session\_id`. \* Invoque o `runner` programaticamente com o prompt desejado e processe o stream de eventos para exibir a resposta final do agente.

═══════════════════════════════════════════════════════

5 VALIDAÇÃO E PONTOS DE CONTROLE ✅

═══════════════════════════════════════════════════════

CHECKLIST DE QUALIDADE

- \*\*Consistência na Passagem de Estado\*\* 🔄: Verifique o uso correto do `output\_key` para assegurar que os dados gerados por um sub-agente (e.g., o roteiro) sejam devidamente acessíveis e utilizados pelos sub-agentes subsequentes (visualizer, formator), mantendo a coerência do fluxo de trabalho.  
- \*\*Cobertura Completa dos Sub-agentes\*\* ✅: Assegure que, ao empregar o `Loop Agent`, todos os sub-agentes definidos para o fluxo de trabalho sejam invocados e executados de forma iterativa, conforme visualizado e confirmado na UI do `ADK web`.  
- \*\*Interoperabilidade e Agnóstico ao Modelo/Implantação\*\* 🌐: Confirme que o ADK suporta a integração de \*qualquer modelo LLM\* e permite a implantação em \*qualquer provedor de nuvem\*, validando a flexibilidade e adaptabilidade prometidas pela estrutura.  
- \*\*Capacidades de Depuração na UI\*\* 🐞: Utilize a interface `ADK web` para monitorar eventos internos e transferências de controle entre agentes, facilitando a depuração e o entendimento aprofundado do fluxo de execução do sistema multi-agente.  
- \*\*Execução Programática Robusta\*\* ⚙️: Verifique a capacidade de invocação programática do agente, garantindo o uso adequado dos `Services` (memória, sessão, artefatos), do `Runner` e do tratamento de `Events` para gerenciar eficientemente o ciclo de vida da interação e os resultados obtidos.

═══════════════════════════════════════════════════════

6 SOLUÇÃO DE PROBLEMAS ⚠️

═══════════════════════════════════════════════════════

PROBLEMAS COMUNS E SUAS SOLUÇÕES

═══════════════════════════════════════════════════════

7 PERGUNTAS FREQUENTES (FAQ) ❓

═══════════════════════════════════════════════════════

═══════════════════════════════════════════════════════

8 CONSIDERAÇÕES FINAIS 📄

═══════════════════════════════════════════════════════

Este manual foi desenvolvido com base nas melhores práticas da Bosch

para garantir padronização e qualidade nos processos organizacionais.

Para sugestões de melhoria ou dúvidas adicionais, entre em contato

com a área responsável: Apresentado por especialistas do \*\*Google ADK\*\*..

Mantenha este manual atualizado e compartilhe com a equipe para

garantir a execução correta dos procedimentos descritos.

═══════════════════════════════════════════════════════

DOCUMENTO GERADO AUTOMATICAMENTE PELO SISTEMA AIDO v3

Data de Criação:

Versão do Sistema: 3.0 ADK

Responsável Técnico: Apresentado por especialistas do \*\*Google ADK\*\*.

© 2025 Robert Bosch GmbH. Todos os direitos reservados.

═══════════════════════════════════════════════════════