□ 100 Q&A on tool_use_behaviour

Basics

1.

   Q: What is tool_use_behaviour?
   A: It defines how an agent selects, prioritizes, and invokes tools during execution.

2.

   Q: Why is tool_use_behaviour important?
   A: It ensures the agent uses tools in a controlled and predictable way.

3.

   Q: Is tool_use_behaviour configurable?
   A: Yes, developers can customize tool invocation strategies.

4.

   Q: Can tool_use_behaviour affect accuracy?
   A: Yes, improper handling may lead to overuse or misuse of tools.

5.

   Q: What's the default behaviour if unspecified?
   A: The agent uses tools automatically when needed.

Tool Invocation Control

6.

   Q: How can you force a tool to always be used?
   A: By setting tool_use_behaviour = "force".

7.

Q: What does "auto" mean in tool_use_behaviour?
A: The agent decides whether to call a tool or not.

8.

Q: What does "none" mean?
A: The agent is restricted from calling any tool.

9.

Q: Can you allow partial tool usage?
A: Yes, by selectively enabling certain tools.

10.

Q: What is "manual" tool use behaviour?
A: It requires explicit developer instructions to trigger tools.

Strategies

11.

Q: What is a greedy strategy in tool use?
A: The agent always calls the first matching tool.

12.

Q: What is a selective strategy?
A: The agent picks the most relevant tool based on context.

13.

Q: What is a fallback strategy?
A: The agent tries one tool and falls back to another if it fails.

14.

Q: What is a sequential strategy?
A: The agent executes multiple tools in order.

15.

Q: What is a parallel strategy?
A: The agent calls multiple tools simultaneously.

Developer Overrides

16.

Q: Can a developer override tool behaviour?
A: Yes, by customizing tool_use_behaviour.

17.

Q: How do you disable one tool only?
A: By setting is_enabled = false for that tool.

18.

Q: Can developers force structured tool outputs?
A: Yes, by defining strict schemas.

19.

Q: How can logging improve tool behaviour?
A: It helps debug unnecessary or failed tool calls.

20.

Q: What role does configuration play in tool use?
A: It enforces limits and prevents misuse.

Safety & Guardrails

21.

Q: Why add guardrails to tool_use_behaviour?
A: To prevent dangerous or excessive tool calls.

22.

Q: Can tools be blocked based on input type?
A: Yes, by using input filters.

23.

Q: What if a tool call exposes sensitive data?
A: Guardrails can block execution.

24.

Q: Can tool calls be rate-limited?
A: Yes, to avoid overload.

25.

Q: How does tool_use_behaviour relate to user trust?
A: Transparent, safe tool usage increases reliability.

Error Handling

26.

Q: What happens if a tool fails?
A: The agent follows fallback or error recovery behaviour.

27.

Q: Can failure trigger retries?
A: Yes, configurable retry logic can be added.

28.

Q: What is a silent failure in tool behaviour?

A: The agent ignores errors without informing the user.

29.

Q: Why should silent failures be avoided?
A: They reduce transparency and cause unexpected results.

30.

Q: Can tool use behaviour log failures automatically?
A: Yes, tracing can capture tool-level errors.

Advanced Configurations

31.

Q: Can tools be prioritized?
A: Yes, tools can have ranking scores.

32.

Q: Can agents dynamically enable/disable tools?
A: Yes, via adaptive tool behaviour.

33.

Q: What is “context-aware” tool behaviour?
A: Tools are used based on user intent and history.

34.

Q: What is “role-based” tool behaviour?
A: Certain tools are available only for specific roles.

35.

Q: Can tool behaviour differ across sessions?
A: Yes, based on session context.

Example Use Cases

36.

Q: Example of forcing calculator tool?
A: Always route numeric queries to the calculator tool.

37.

Q: Example of fallback tool?
A: If API A fails, use API B.

38.

Q: Example of parallel tools?
A: Query both weather API and location API at the same time.

39.

Q: Example of selective tool use?
A: Use translation tool only if input is non-English.

40.

Q: Example of restricted tool use?
A: Block database queries from unverified users.

Execution Flow

41.

Q: When does an agent decide to call a tool?
A: During reasoning, based on configured behaviour.

42.

Q: How does priority influence tool calls?
A: Higher-priority tools are checked first.

43.

Q: Can agents skip tools entirely?
A: Yes, if behaviour allows.

44.

Q: What is a tool invocation cycle?
A: The sequence of tool calls during a run.

45.

Q: Can behaviour enforce a maximum number of tool calls?
A: Yes, to control cost and time.

Integration with Other Features

46.

Q: How does tool_use_behaviour interact with handoff?
A: Tools can be invoked before or after handing off tasks.

47.

Q: How does it interact with context?
A: Context guides tool selection.

48.

Q: How does it interact with runner?
A: Runner enforces limits on tool calls.

49.

Q: How does it interact with guardrails?
A: Guardrails restrict tool misuse.

50.

Q: How does it interact with structured output?
A: Tool outputs can follow schema definitions.

Monitoring & Tracing

51.

Q: Why trace tool behaviour?
A: For debugging and optimization.

52.

Q: Can logs show tool call frequency?
A: Yes, through telemetry.

53.

Q: Can tracing capture tool failures?
A: Yes, with detailed error reports.

54.

Q: What is anomaly detection in tool behaviour?
A: Identifying abnormal usage patterns.

55.

Q: Can tool behaviour be audited?
A: Yes, for compliance and trust.

Scalability

56.

Q: Why is tool behaviour critical in multi-agent systems?
A: It prevents conflicts when multiple agents call tools.

57.

Q: Can tool usage be load-balanced?
A: Yes, by distributing calls across replicas.

58.

Q: Can behaviour prevent API overload?
A: Yes, with throttling.

59.

Q: Can behaviour optimize cost?
A: Yes, by avoiding unnecessary tool calls.

60.

Q: Can tool use be scheduled?
A: Yes, to manage resources better.

Security

61.

Q: Can tool behaviour block malicious inputs?
A: Yes, via filters.

62.

Q: Can unauthorized tools be blocked?
A: Yes, using permission rules.

63.

Q: Can tool results be sanitized?

A: Yes, before returning to the agent.

64.

Q: What’s the risk of unsafe tool behaviour?
A: Data leakage or unintended actions.

65.

Q: Can tool use be logged for audits?
A: Yes, securely.

Adaptive Behaviour

66.

Q: What is adaptive tool behaviour?
A: Changing tool strategy dynamically.

67.

Q: Can agents learn from failed tool calls?
A: Yes, with adaptive logic.

68.

Q: Can user preferences adjust tool behaviour?
A: Yes, by saving personalized settings.

69.

Q: Can tool behaviour evolve with context?
A: Yes, with session memory.

70.

Q: Can feedback loops refine behaviour?
A: Yes, by analyzing outcomes.

Optimization

71.

Q: How can tool usage be optimized?
A: By reducing redundant calls.

72.

Q: What is caching in tool use?
A: Storing results to avoid repeat calls.

73.

Q: How does batching help?
A: Multiple queries are sent together.

74.

Q: What is deduplication?
A: Preventing duplicate tool calls.

75.

Q: How can costs be reduced?
A: By enforcing tool budgets.

Testing

76.

Q: Why test tool behaviour?
A: To ensure reliability.

77.

Q: What is a unit test for tool use?
A: Checking if a tool is invoked correctly.

78.

Q: What is integration testing?
A: Validating multiple tools working together.

79.

Q: Can simulation test tool failures?
A: Yes, by injecting errors.

80.

Q: Can behaviour be stress-tested?
A: Yes, with high-load testing.

Real-World Applications

81.

Q: Example in healthcare?
A: Only allow approved diagnostic tools.

82.

Q: Example in finance?
A: Use fallback pricing APIs.

83.

Q: Example in education?
A: Restrict internet tools during exams.

84.

Q: Example in customer service?
A: Prioritize FAQ tool before escalating.

85.

Q: Example in logistics?
A: Use real-time traffic APIs adaptively.

Edge Cases

86.

Q: What if no tool is available?
A: The agent continues without tools.

87.

Q: What if all tools fail?
A: The agent returns a safe fallback response.

88.

Q: What if tool output is invalid?
A: Apply schema validation.

89.

Q: What if multiple tools conflict?
A: Use priority or arbitration rules.

90.

Q: What if tool returns sensitive info?
A: Sanitize before returning.

Future Directions

91.

Q: Will tool behaviour become more autonomous?
A: Yes, with self-learning strategies.

92.

Q: Can tools use reinforcement learning?
A: Yes, for adaptive decisions.

93.

Q: Will behaviour include ethical constraints?
A: Yes, to enforce safety.

94.

Q: Can tools negotiate with each other?
A: Yes, in multi-agent systems.

95.

Q: Will AI governance regulate tool use?
A: Likely, for accountability.

Final Wrap-Up

96.

Q: What’s the biggest risk in tool behaviour?
A: Uncontrolled or malicious tool execution.

97.

Q: What’s the biggest benefit?
A: Efficiency and accuracy.

98.

Q: How does it help developers?

A: Provides fine control over agent actions.

99.

Q: How does it help users?
A: More reliable and safe outputs.

100.

Q: One line definition?
A: tool_use_behaviour defines how, when, and why agents use tools.