

ИИ-агенты и Agent Skills

Методическое руководство для учителя

Редакция r0002 · 2026-10-14 · starter 0.17.2

Техническая редакция: пилот не проводился

ИИ-агенты и Agent Skills

Методическое руководство для учителя

Редакция	r0002
Дата сборки	2026-10-14
starter	0.17.2

Техническая редакция: пилот не проводился, живых свидетельств модели нет (education_status: pre-pilot, publication_ready: false).

Состав

Часть	Версия
Методическое руководство для учителя	1.9
Поурочная карта	1.4

Оглавление

Методическое руководство для учителя	5
0. Как пользоваться этим руководством	5
1. Для кого курс и какой результат ожидается	6
2. Два расписания: 34 и 18 часов	6
3. Треки А и В	8
4. Какая «модель» работает на уроках и что считать доказательством	8
5. Как устроены главы и контрольные точки	9
6. Конструктор «Дежурного»: что ученик накапливает и как это проверить	11
7. Безопасность практикума и Урок 0	15
8. Что ещё не проверено и как учитель помогает пилоту	15
Поурочная карта.....	17
Урок Э1. Модель и два входа к ней.....	17
Урок Э2. Агент и его упряжка	18
Урок 1. Что скрывается за одним окном чата	18
Урок 2. Когда помощник начинает выбирать	19
Урок 3. Один шаг и обратная связь	20
Урок 4. Четыре способа завершить запуск	20
Урок 5. Кто организует знакомый цикл	21
Урок 6. Что видно сейчас и что действительно разрешено	21
Урок 7. Почему пять похожих слов обозначают разные механизмы.....	22
Урок 8. Как механизмы работают вместе	22
Урок 9. От просьбы к договору задачи	23
Урок 10. Когда агент не нужен.....	23
Урок 11. Как модель запрашивает tool	24
Урок 12. Параметры как договор	24
Урок 13. Диспетчер и белый список.....	25
Урок 14. Ошибки, несколько вызовов и возврат результата.....	25
Урок 15. Хороший tool начинается с границы	26
Урок 16. Validator и три обязательных теста	26
Урок 17. Что такое настоящий Agent Skill	27
Урок 18. Пишем SKILL.md	27
Урок 19. Состояние, память, RAG и журнал — не одно и то же	28
Урок 20. Как передать работу следующему запуску	28
Урок 21. От теста к Eval	29
Урок 22. Сравниваем две версии на одном наборе	29
Урок 23. Trace: что происходило на самом деле	30
Урок 24. Ищем ошибку по Trace	30
Урок 25. Как поправка превращается в опыт	31
Урок 26. Version, Regression и Rollback.....	31
Урок 27. Модель угроз и минимальные полномочия.....	32
Урок 28. Человек внутри цикла	32
Урок 29. Prompt injection внутри обычного документа.....	33
Урок 30. Аудит чужого skill и MCP-подключения	33

Урок 31. Один проект — несколько запусков	34
Урок 32. Один агент, несколько агентов или человек.....	34
Урок 33. Интеграция: проект как единая система	35
Урок 34. Публичная защита и инженерная рефлексия.....	36

Методическое руководство для учителя

0. Как пользоваться этим руководством

Руководство написано для учителя информатики или математики, который ведёт курс впервые. Специальной подготовки по машинному обучению не нужно: всё, что требуется на уроках, есть в книге ученика и в учебном комплекте программ starter. Ссылки вида «с. 39» ведут на страницы книги ученика.

В руководстве четыре части: разделы 1–8 — как устроен курс и как его вести; поурочная карта — что происходит на каждом из 34 уроков; конспект глав — одна страница на главу для быстрой подготовки; список фигур — где в книге стоит каждая схема.

Ответы. Ответы на мини-проверки уроков и на вопросы «Проверь себя» напечатаны в самой книге — в разделах «Ответы» сразу после вопросов. Исключение — «Проверь себя» глав 1 и 2: напечатанных ответов там нет. Ответы к главе 1 сверяются по уроку 2 (–), а к главе 2 — по урокам 3 и 4 (–); ответы мини-проверки ролей — . Ответ на вопрос-прогноз, с которого начинается глава, — в её конце, где класс возвращается к сцене. Отдельного сборника ответов нет: курс строится на том, что ученик сначала записывает прогноз, а потом проверяет его, поэтому ответ не сообщают раньше механизма.

Слова, которые встретятся в руководстве

Точные определения — в Приложении Б книги ученика (). Здесь — коротко, чтобы читать руководство.

Слово	Что это простыми словами	Где в книге
«Дежурный»	учебный проект курса: программа-помощник школьного проектного бюро, которая читает карточки задач и готовит черновик отчёта о состоянии проекта; ученик настраивает её от главы к главе	гл. 1,
starter	учебный комплект: папка с программами на Python, учебными данными и проверками; все команды курса выполняются из неё	Приложение А,
агентный цикл (agent loop)	повторение «состояние → выбор действия → выполнение → наблюдение → решение»; решение определяет, продолжать или остановиться	гл. 2,
терминал	итог запуска: success (готово), needs_input (нужен ответ человека), error (ошибка), limit (исчерпан предел шагов)	гл. 2,
tool	конкретная операция, которую программа умеет выполнять, например прочитать карточки задач	гл. 1, 4, 6
harness	программная среда вокруг модели: организует цикл, выполнение действий, состояние, ограничения, ошибки и участие человека	гл. 3,
Agent Skill	переносимая процедура, записанная в файле SKILL.md, с дополнительными файлами при необходимости	гл. 4 и 9,
заглушка (stub)	маленькая функция на месте модели с заранее заданным поведением	гл. 1–5
replay (воспроизведение)	ответы модели, подготовленные заранее; это сценарий, а не запись живой модели	гл. 2 и 6
bridge-сеанс	единственный обязательный ручной сеанс с живой моделью: человек переносит подготовленный запрос в разрешённый чат и возвращает ответ на проверку	гл. 7,

Слово	Что это простыми словами	Где в книге
trace	сохранённая запись событий одного запуска: что произошло, кем начато, чем закончилось	гл. 12,
test case, eval	заранее заданная ситуация с ожидаемым результатом; eval — проверка поведения на наборе таких ситуаций	гл. 8 и 11
approval gate	место, где программа останавливается и ждёт решения человека	гл. 14,
checkpoint, handoff	сохранённое состояние, по которому работу можно продолжить, и передача работы следующему запуску или человеку; не путать с контрольной точкой курса (КТ)	гл. 10 и 16
Release candidate	версия проекта, зафиксированная для итоговых проверок перед защитой	гл. 17,
golden	файл с эталонным выводом программы; по нему проверяется, что программа печатает то же, что книга	раздел 4 руководства

1. Для кого курс и какой результат ожидается

Курс рассчитан прежде всего на учащихся 9–10 классов. Он не требует вузовской подготовки по машинному обучению, коммерческого API-ключа, сложной инфраструктуры или уверенного владения Python. Ученик рассматривается как начинающий инженер: он должен различать роли частей системы, читать простые схемы, проектировать ограничения, сравнивать ожидание с наблюдением, находить ошибку и аргументировать решение.

Образовательный маршрут строится как последовательность наблюдаемых действий: понять устройство системы, смоделировать её, собрать учебную версию, проверить, ограничить последствия, улучшить и объяснить другому. Итогом является локальный проект «Дежурный» — программа-помощник школьного проектного бюро, которую ученик настраивает от главы к главе, — и его защита. От ученика не требуется собрать профессиональную автономную систему; требуется показать связную архитектуру, корректные границы, проверяемые результаты и умение объяснить, какие утверждения подтверждены практикой, а какие остаются предположениями.

Руководство не является ключом к заданиям. Сцены, прогнозы и спорные вопросы следует проходить с классом в том порядке, в каком они даны в главах. Учителю важно не сообщать ответ раньше механизма: ценность курса во многом состоит в том, что ученик сначала делает прогноз, а затем сталкивает его с наблюдаемым результатом.

2. Два расписания: 34 и 18 часов

Полный маршрут занимает 34 академических часа: 17 глав по два урока. Урок 0 вынесен в Приложение А и является организационной подготовкой рабочего места, а не 35-м часом. Перед главой 1 стоит вводная глава «Экскурсия по AI: что есть что» — два урока-показа Э1–Э2 сверх 34 часов (итого 36): ученик предсказывает, наблюдает и заполняет таблицу наблюдений, практик со starter и шага проекта в них нет; карточки — в поурочной карте (,). В сокращённом маршруте Экскурсия сжимается до одного урока-показа или выносится в домашний просмотр (строка «Э» таблицы ниже). В полном маршруте главы идут последовательно: от различения модели, приложения и агента к циклу, harness, механизмам взаимодействия, договору задачи, tools, Agent Skill, состоянию, проверкам, traces, версиям, безопасности, долгой работе и итоговой защите.

Сокращённый маршрут занимает 18 занятий по 45 минут. Он объединяет пары уроков, но сохраняет главные понятия, безопасность, проектирование и защиту; какие уроки в какое занятие входят — в таблице ниже. Нельзя исключать точные роли, остановку, договор задачи, tool, SKILL.md, состояние вне чата, три типа проверок, trace, минимальные полномочия и итоговую защиту. Личный skill и реальный bridge-сеанс для трека В также остаются обязательными.

В расширение можно вынести прямое подключение программы к модели через API, настоящее MCP-подключение, автоматический запуск проверок (CI), сложную автоматизацию, дополнительные реализации и часть углубляющих инженерных упражнений. Такое сокращение не должно превращать курс в демонстрацию готового кода. Даже в 18-часовой версии ученик проходит путь «спрогнозировал → проверил → объяснил границу» и предъявляет наблюдаемый продукт.

Сокращённый маршрут: 18 занятий по 45 минут

Занятие	Уроки полного курса	Обязательное	Расширение	с. в книге
Э	Э1–Э2	Один урок-показ: показы 1, 4, 5, 7 (только три роли), 8 и карта «что есть что»; показы 2, 3, 6 — домашний просмотр видео или чтение. Либо вся Экскурсия — домашним просмотром, тогда первые 5 минут занятия 1 — сверка таблиц наблюдений	Второй урок-показ	,
1	1–2	Роли, одинаковая фраза и различное наблюдение	Правка заглушки	,
2	3–4	4 стадии + решение, 4 терминала; показ учителя в конце главы 2	Пять программных сценариев	,
3	5–6	Harness, workspace, состояние и права	Реализация проверки пути	,
4	7–8	Пять механизмов; контроль зависимостей	Учебная MCP-карточка	,
5	9–10	Договор задачи; когда агент не нужен	Дополнительные кейсы	,
6	11–12	Чтение без пути от модели; task_id; schema; подготовка bridge	Сложные схемы	,
7	13–14	Allowlist, retry/error; реальный bridge минимум В	Дополнительный сеанс	,
8	15–16	Паспорт tool и три проверки	Автовалидатор	,
9	17	Каталог Agent Skill: описание, инструкции, ресурсы	Лицензия и расширенные поля	
10	18	Проектный skill и личный skill v1; три проверки	Script после аудита	
11	19–20	project_state, handoff, отсутствие секретов	RAG-демо	,
12	21–22	Harness отдельно от model_behavior; критерии	Автоматический eval-runner	,
13	23–24	Trace: инициатор события, поиск сбоя	Анализатор trace	,
14	25–26	Личный skill v2, diff, перпексия, keep/revert (КТ 7)	Git	,
15	27–28	Минимальные права, read-only, approval	Полная матрица угроз	,

Занятие	Уроки полного курса	Обязательное	Расширение	с. в книге
16	29–30	Инъекция и аудит стороннего skill/MCP	Статический сканер	,
17	31–32	Возобновление и выбор один/несколько/человек	Запуск нескольких агентов	,
18	33–34	Интеграция, защита, bridge-доказательство В, личный skill	Развёрнутая рефлексия	,

Маршрут	Что сохраняется обязательно	Что можно оставить расширением
34 часа	17 глав, КТ 1–8, три проверочные работы, интеграция, защита	дополнительные реализации и факультативные сервисы
18 часов	понятийное ядро, безопасность, личный skill, bridge В, trace, версии, защита	API/live, настоящий MCP, CI, сложная автоматизация

При планировании стоит учитывать, что хронометраж отдельных сжатых занятий ещё требует классной проверки. Поэтому 18-часовой маршрут следует вести с запасом на обсуждение ошибок и не компенсировать нехватку времени пропуском безопасности или итогового доказательства.

3. Треки А и В

Трек А и трек В равноправны. Трек А использует карточки, схемы, ролевое моделирование, проектирование, анализ и защиту без обязательного программирования. Трек В использует Python и локальный starter. Различается форма практической реализации, но архитектурное понимание и безопасность оцениваются одинаково.

Если на уроке возникает технический сбой, ученик не должен терять содержательную часть занятия. Он начинает с трека А, фиксирует прогноз, строит схему и разбирает границы, а школа организует доступ к треку В позже. Инструкции по установке и подготовке среды не нужно переписывать в это руководство: единственный источник — Приложение А «Урок 0. Готовность рабочего места».

Учителю полезно проверять не объём написанного кода, а качество инженерного объяснения. Ученик должен уметь показать, где выполняется действие, какой результат наблюдаем, что ограничивает последствия и почему один удачный запуск не доказывает устойчивость всей системы. Трек В добавляет программное свидетельство, но не отменяет необходимость объяснения.

4. Какая «модель» работает на уроках и что считать доказательством

Режимы модели в курсе: stub — заглушка в обязательных практиках глав 1–5 (в главе 5 модель в практике не участвует); replay — авторский сценарий с указанным происхождением, с главы 6 в практиках и с главы 1 в конструкторе «Дежурного»; bridge — единственный обязательный ручной сеанс с живой моделью (глава 7), его свидетельство — на защите; live/API — факультатив. Replay не оценивает качество живой модели; bridge не закрывается заранее подготовленным ответом.

Режим	Что это	Где в курсе	Что можно утверждать по результату
заглушка (stub)	функция с заранее заданным поведением на месте модели	практики глав 1–5	программа вокруг модели работает так, как описано
replay	ответы, подготовленные заранее	практики с главы 6; «Дежурный» с главы 1	программа правильно обрабатывает такой ответ; о живой модели — ничего
bridge	человек переносит запрос в разрешённый чат с живой моделью и возвращает ответ на проверку	один раз, глава 7 (); свидетельство — на защите	этот конкретный ответ живой модели прошёл проверки программы
live/API	прямое подключение программы к модели	факультатив	в обязательную часть курса не входит

Это различие нужно проговаривать всякий раз, когда класс видит убедительный текстовый результат. Заглушка и replay позволяют воспроизводить учебные ситуации и проверять harness, формат данных, правила остановки, обработку ошибок и настройки проекта. Они не дают основания утверждать, что живая модель в похожей ситуации ответит так же.

Bridge устроен иначе: код не обязан подключаться к сети. Harness готовит контекст, человек переносит его в доступный живой чат и возвращает структурированный ответ, который проверяется до выполнения. Для курса важен не эффект «мы поговорили с ИИ», а паспорт реального сеанса: происхождение ответа, условия запуска и наблюдаемое продолжение работы. Заранее известный эталонный вывод (в starter такие файлы называются golden) не может быть доказательством такого сеанса.

Платный API и личная регистрация ученика не обязательны. Доступный живой чат для обязательного сеанса организует школа или учитель с соблюдением правил выбранного сервиса. Если такого доступа в момент урока нет, класс продолжает офлайн, а live-доказательство остаётся открытым до организованной школьной сессии. Для сеанса подойдёт любой чат с живой моделью, которым школа разрешает пользоваться. Что перенести в чат и что вернуть, описано в главе 7 (); итог записывается в паспорт по шаблону `workspace/ch07/bridge_passport_template.md` ().

5. Как устроены главы и контрольные точки

Рабочий ритм главы стабилен: сцена проектного бюро → вопрос-прогноз → объяснение механизма → «Проверь себя на ходу» → практика → шаг сквозного проекта → возврат к сцене → мост к следующему вопросу. Эта повторяемость помогает ученику отличать содержание главы от формы работы. Учителю полезно сохранять прогнозы до запуска: после практики их можно сравнить с результатом и увидеть не только ошибку программы, но и ошибку в рассуждении.

Контрольные точки курса привязаны к наблюдаемым продуктам. «КТ курса» — это организационная контрольная точка учебного маршрута; её не следует смешивать с техническим Checkpoint, который появляется позже как часть продолжения долгой работы.

Урок	Контроль	Что сдаёт ученик	с. в книге
4	КТ 1: цикл	программа проходит пять учебных сценариев; получены все четыре итога запуска; ошибка — на втором действии	
8	КТ 2: архитектура	карта системы: роль каждого механизма и связи между ними	

Урок	Контроль	Что сдаёт ученик	с. в книге
14	КТ 3: цикл и bridge	чтение, ошибка и результат в одном запуске; в треке В — паспорт bridge-сеанса	
16	Проверочная работа 1	tool, схема параметров (schema), проверка (validator), три теста	
18	КТ 4: skill	skill проекта и личный skill; три проверки	
20	КТ 5: состояние	передача работы между запусками (handoff)	
22	Проверочная работа 2	проверка программы отдельно от поведения модели; исходный уровень (baseline)	
24	КТ 6: диагностика	отчёт об инциденте по trace	
26	КТ 7: версии	личный skill v1 и v2, сравнение (diff), повтор случаев, решение keep или revert	
28	Проверочная работа 3	разрешения (permissions) и approval gate	
30	КТ 8: аудит доверия	внешний аудит skill другой группы: обоснованный отказ или условия подключения	
33	Интеграция	кандидат проекта для итоговой проверки	
34	Итог	защита, свидетельство bridge-сеанса (трек В), личный skill; рубрика —	

На защите ценится не количество функций, а связность доказательств. Ученик должен показать архитектуру, договор задачи, качество tools и skill, состояние, проверки, trace, безопасность и решение человека там, где оно требуется. Критическое нарушение — например, произвольное выполнение команды, выход за учебную папку, хранение секрета или значимое изменение без нужного подтверждения — не компенсируется хорошей презентацией. Стобалльная рубрика защиты и правило бонуса приведены в главе 17 () — ученик видит те же критерии.

Компонент	Баллы
Понимание архитектуры	15
Постановка задачи и критерий готовности	10
Качество tools	10
Качество Agent Skill	10
Состояние и возобновление	10
Тесты и evals	15
Trace и обработка ошибок	10
Безопасность и подтверждения	15
Документация и защита	5
Итого	100

Бонус до 5 баллов сверх 100: 2 — измеримое улучшение личного skill на неизменном наборе; 2 — перенос процедуры на другой учебный пример; 1 — воспроизводимые diff/версия/keep-revert. Отчётность: основной результат /100 и бонус /5 отдельно. Бонус не компенсирует нарушения безопасности и не заменяет обязательный bridge-сеанс в треке В.

6. Конструктор «Дежурного»: что ученик накапливает и как это проверить

Ученик не пишет заново весь harness. По ходу курса он добавляет настройки в `workspace/my_agent/`, а «Дежурный» показывает, как они влияют на один и тот же локальный проект. В главах 4 и 6 отдельного файла настройки нет: это намеренно, потому что не каждая тема курса должна превращаться в новую строку конфигурации.

Глава	Настройка ученика	Что наблюдает учитель
1	<code>01_goal.md</code>	цель в строке <code>goal:</code>
2	<code>02_stop_policy.json</code>	предел шагов и повторов
3	<code>03_workspace_policy.json</code>	разрешённая область записи; отказ вне неё
4	отдельного файла нет	понятийная глава; фиктивная настройка не создаётся
5	<code>05_task_contract.md</code>	<code>contract: ok/violated</code> по «Готово, когда»
6	отдельного файла нет	вызов операции входит в общий harness
7	<code>07_tools.json</code>	разрешённый список; неизвестный tool отклоняется до диспетчера
8	<code>08_my_tool.py</code>	имя и описание функции; программа читает текст файла, но не запускает его код
9	<code>09_skills/<имя>/SKILL.md</code>	skill ученика в каталоге; загрузка тремя слоями; три проверки главы 9
10	<code>10_state.json</code>	второй запуск видит сохранённое состояние
11	<code>11_evals.json</code>	случаи ученика и итог <code>result: n/m</code>
12	<code>12_trace.json</code>	запись trace и разбор последнего запуска
13	<code>13_versions/</code>	снимки конфигурации и сравнение без автоматического изменения файлов
14	<code>14_permissions.json</code>	gate закрыт по умолчанию; изменение не применяется без решения человека
15	<code>15_trust.json</code>	решение аудита для skills; подключение не разрешает scripts автоматически
16	<code>16_run_plan.json</code>	одна команда <code>run</code> = одна Session; предел числа сессий
17	<code>17_release_candidate.json</code>	фиксация одной версии для итоговых проверок; метка не доказывает качество

Счётчик `configured: k/15` относится к пятнадцати реально настраиваемым строкам. Для быстрой проверки удобно идти в порядке `status` → `run` → `eval`, а при разборе конкретного запуска — к последнему `trace`. Файлы `08_my_tool.py`, `09_skills/.../SKILL.md` и `05_task_contract.md` дополнительно требуют ручного просмотра по правилам соответствующих глав.

Все команды набираются в командной строке, открытой в папке `starter` — той, где лежит файл `dezhurny.py`. Строка `# запуск из корня starter` в примерах напоминает об этом; набирать её не нужно. В примерах написано `python`; на Windows обычно набирают `py -3`, на macOS и Linux — `python3`, как в Приложении А ().

Прежде чем смотреть настройки учеников, проверь целостность самой поставки — на свежеустановленном `starter`, до первого запуска. Команда рассчитана на нетронутый комплект: на папке, где уже лежат `trace`, черновики или файлы учеников в `my_agent/`, она честно ответит `false`, хотя `starter` исправен.

```
# запуск из корня starter
python scripts/verify_release.py
```

Команда печатает отчёт в формате JSON и занимает около десяти секунд; смотреть нужно на строку "all_passed": true в конце: starter цел, тесты проходят; о качестве модели это ничего не говорит. Полный вывод здесь намеренно не приводится, потому что технические счётчики меняются вместе с версией starter.

Держите две копии starter. Первая — чистая: её проверяют командой `verify_release.py` и с неё раздают комплект ученикам; в ней ничего не запускают. Вторая — рабочая: на ней учитель повторяет примеры этого раздела и смотрит работы учеников. Команды `run` и `trace` записывают в папку файлы — черновики и `trace`, поэтому после них `verify_release.py` на той же папке ответит `false`. Это не поломка.

Первая команда показывает состав настроек и счётчик. Смотрите на третий столбец (настроено / не настроено / ошибка) и на последнюю строку `configured: k/15`. Ниже — проверенный пример для базовой учебной конфигурации.

```
# запуск из корня starter
python dezhurny.py status
```

```
dezhurny: status
config: workspace/my_agent
глава 1 | 01_goal.md | не настроено | run: заголовок цитирует первую строку цели
глава 2 | 02_stop_policy.json | не настроено | run: предел шагов и повторов
глава 3 | 03_workspace_policy.json | не настроено | run: сохранение черновика только в
разрешённую область
глава 5 | 05_task_contract.md | не настроено | run: contract ok/violated по разделу
«Готово, когда»
глава 7 | 07_tools.json | не настроено | run: tool вне списка отклоняется до диспетчера
глава 8 | 08_my_tool.py | не настроено | каталог tools: описание функции ученика
глава 9 | 09_skills | не настроено | каталог skills: skill ученика, три слоя загрузки
глава 10 | 10_state.json | не настроено | run: второй запуск видит первый
глава 11 | 11_evals.json | не настроено | eval: таблица случаев ученика
глава 12 | 12_trace.json | не настроено | run: trace в workspace/traces/dezhurny/; trace --
last
глава 13 | 13_versions | не настроено | version save сохраняет снимок конфигурации
глава 14 | 14_permissions.json | не настроено | approval gate закрыт: изменение статуса не
применяется
глава 15 | 15_trust.json | не настроено | skills из 09_skills подключаются без записанного
решения аудита
глава 16 | 16_run_plan.json | не настроено | run — одна Session без checkpoint
глава 17 | 17_release_candidate.json | не настроено | run: печатает метку Release
candidate; изменение любого файла после фиксации делает её недействительной
configured: 0/15
```

Этот вывод доказывает, какие строки конфигурации распознаны и каков счётчик. Он не доказывает, что полный запуск успешен, что все ручные артефакты качественны или что живая модель работает хорошо.

Следующая команда показывает техническую границу записи на заранее подготовленном случае.

```
# запуск из корня starter
python dezhurny.py run --dir workspace/my_agent/examples/denied
```

```

dezhurny: run
goal: не настроено (глава 1)
config: workspace/my_agent/examples/denied
model: replay – авторский сценарий, не запись модели
scenario: success
stop_policy: max_steps 5, max_retries 1 (по умолчанию, глава 2)
tools: allowlist диспетчера (по умолчанию, глава 7)
Шаг 1 | read_task_cards | continue
Шаг 2 | validate_task_card | continue
Шаг 3 | validate_task_card | continue
Шаг 4 | validate_task_card | continue
Шаг 5 | build_status_report | success
Остановка: success
# Черновик статуса проекта

Всего задач: 3.
Незавершённых: 2.
Заблокированных: 1.

## Незавершённые задачи
- `task-002` – Подготовить макет афиши (in_progress)
- `task-003` – Проверить список оборудования (blocked)

## Блокеры
- `task-003`: Нет подтверждённого списка оборудования

Черновик. Отправка и изменение карточек не выполнялись.
contract: не настроено (глава 5)
save: denied | path_not_writable | tasks/dezhurny_report.md
state: не настроено (глава 10)
trace: не настроено (глава 12)
Качество модели не подтверждено этим техническим прогоном.

```

Здесь наблюдаемо доказан отказ записи вне разрешённой области. Этот прогон не доказывает наличие системной изоляции уровня ОС и сам прямо не подтверждает качество модели.

Ограничение числа шагов можно проверить отдельным подготовленным случаем.

```

# запуск из корня starter
python dezurny.py run --dir workspace/my_agent/examples/limit2

```

```

dezhurny: run
goal: не настроено (глава 1)
config: workspace/my_agent/examples/limit2
model: replay – авторский сценарий, не запись модели
scenario: success
stop_policy: max_steps 2, max_retries 1
tools: allowlist диспетчера (по умолчанию, глава 7)
Шаг 1 | read_task_cards | continue
Шаг 2 | validate_task_card | limit
Остановка: limit
contract: не настроено (глава 5)
save: не настроено (глава 3)
state: не настроено (глава 10)
trace: не настроено (глава 12)
Качество модели не подтверждено этим техническим прогоном.

```

Этот вывод доказывает, что настройка предела меняет наблюдаемый терминал на `limit`. Он не доказывает, что выбранный предел оптимален для всех задач.

Для проверки набора случаев используется `eval`.

```
# запуск из корня starter
python dezturny.py eval --dir workspace/my_agent/examples/full
```

```
dezturny: eval
config: workspace/my_agent/examples/full
case report-on-clean-cards: scenario success | expected success | actual success | pass
case ask-on-ambiguity: scenario needs_input | expected needs_input | actual needs_input |
pass
case stop-on-missing-cards: scenario missing_file | expected error | actual error | pass
result: 3/3
evidence: synthetic_script – проверка harness и настроек my_agent, не качество модели
```

Результат 3/3 доказывает прохождение именно этих подготовленных случаев для harness и настроек. Он не является оценкой поведения живой модели и не заменяет новые случаи, если ученик изменил границы задачи.

Trace пишется, только если в папке workspace/my_agent/ лежит файл 12_trace.json с текстом {"enabled": true}; без такого файла и без предшествующего запуска команда trace --last честно отвечает, что записанных запусков нет. Чтобы повторить пример ниже, создайте этот файл в рабочей копии starter и выполните две команды подряд.

```
# запуск из корня starter
# фикстура: tests/fixtures/ch12/12_trace.json → workspace/my_agent/12_trace.json
python dezturny.py run
```

```
dezturny: run
goal: не настроено (глава 1)
config: workspace/my_agent
model: replay – авторский сценарий, не запись модели
scenario: success
stop_policy: max_steps 5, max_retries 1 (по умолчанию, глава 2)
tools: allowlist диспетчера (по умолчанию, глава 7)
War 1 | read_task_cards | continue
War 2 | validate_task_card | continue
War 3 | validate_task_card | continue
War 4 | validate_task_card | continue
War 5 | build_status_report | success
Остановка: success
# Черновик статуса проекта
```

Всего задач: 3.
Незавершённых: 2.
Заблокированных: 1.

```
## Незавершённые задачи
- `task-002` – Подготовить макет афиши (in_progress)
- `task-003` – Проверить список оборудования (blocked)
```

```
## Блокеры
- `task-003`: Нет подтверждённого списка оборудования
```

Черновик. Отправка и изменение карточек не выполнялись.
contract: не настроено (глава 5)
save: не настроено (глава 3)
state: не настроено (глава 10)
trace: workspace/traces/dezturny/run-001.jsonl
Качество модели не подтверждено этим техническим прогоном.

```
# запуск из корня starter
python dezurnny.py trace --last
```

```
trace: workspace/traces/dezhurnny/run-001.jsonl
terminal: success
tool_calls: 5
errors: 0
failure_point: нет наблюдаемого отклонения
```

Эта команда доказывает, что в сохранённом trace наблюдаемы терминал, число вызовов, ошибки и место отклонения. Она не объясняет причину сама по себе: если отклонение есть, учитель возвращается к событиям запуска и проверяет цепочку по материалу главы 12.

7. Безопасность практикума и Урок 0

Рабочее место готовит школа. Python, редактор кода, командная строка и starter должны быть доступны до основного занятия; платный API, Git, pip и локальный сервер не являются обязательными входными требованиями. Если подготовка ещё не завершена, содержательная работа начинается в треке А. Инструкции установки и проверки платформы берутся только из Приложения А и здесь не дублируются.

Основные правила практикума действуют с первого урока: синтетические или обезличенные данные, только чтение по умолчанию (read-only), работа внутри учебной папки, отсутствие секретов в коде и данных, отказ от обхода школьных ограничений, решение человека перед значимым изменением. Внешний документ, skill или подключение не становится доверенным только потому, что внутри содержится убедительная инструкция.

Перед занятием учитель проверяет целостность starter командой из начала раздела 6 — только на свежераспакованном комплекте, до первого запуска.

Эта проверка не отменяет педагогическую подготовку. Даже полностью прошедшие тесты не показывают, сколько времени займёт урок, поймут ли ученики формулировку или будет ли живая модель отвечать устойчиво. Для этого нужны отдельные наблюдения в классе и реальный сеанс там, где он предусмотрен.

8. Что ещё не проверено и как учитель помогает пилоту

Программы и тексты курса проверены; не хватает двух свидетельств, которые можно получить только в жизни. Поле `classroom_pilot_pending` означает, что уроки 1–4 ещё должны пройти классный пилот с наблюдением хронометража и нагрузки. Поле `live_model_evidence_pending` означает, что требуется подтверждённый ручной сеанс с живой моделью. Пока оба свидетельства не получены, `publication_ready` остаётся false.

Состояние можно увидеть в кратком отчёте:

```
# запуск из корня starter
python eval_runner.py --summary
```

```
harness: 6/6 passed
model_behavior: not_evaluated (verified_sessions: 0; scripted: 3, excluded from scoring)
publication_ready: false (classroom_pilot_pending, live_model_evidence_pending)
```

Этот вывод доказывает, что технические проверки harness пройдены и что два внешних свидетельства ещё открыты. Он не означает, что главы книги «не готовы», и не позволяет закрыть эти два пункта текстом, replay-сценарием или дополнительным эталонным выводом. Классный пилот должен дать наблюдение за реальными учениками, а bridge-сеанс — реальное свидетельство работы с живой моделью.

Для подготовки к урокам в этом руководстве есть поурочная карта, конспект глав с одной страницей на каждую главу и список фигур со страницами книги. Они помогают быстро увидеть, что делать на уроке, что вводится, что запрещено, какой наблюдаемый продукт ожидается и где в книге расположена схема. Эти материалы не заменяют текст глав и не создают новых требований к ученику.

Перед стартом класса удобно проверить четыре вещи: рабочее место подготовлено по Приложению А; выбран маршрут 34 или 18 часов; понятно, как класс разделится между треками А и В; для обязательного bridge предусмотрен доступный живой чат или заранее назначена школьная сессия. После этого курс можно вести по единому принципу: каждое сильное утверждение должно опираться на наблюдаемое доказательство, а граница этого доказательства проговаривается так же явно, как сам результат.

Если вы ведёте пилотные уроки 1–4, после каждого урока заполните одну страницу: сколько минут ушло на каждый блок урока против плана; что из практик А и В группы успели, а что нет; на чём ученики споткнулись — какая команда, какая строка вывода, какая фраза книги вызвала вопрос. Нужны факты, а не общие впечатления: по ним авторы исправляют книгу.

Поурочная карта

Карта помогает быстро подготовить 34 урока полного маршрута и два урока-показа вводной главы «Экскурсия по AI: что есть что» (Э1–Э2, сверх 34 часов); содержание сверено с поурочным планом и текстом книги, а при расхождении используется формулировка книги.

Урок Э1. Модель и два входа к ней

Страница в книге:

Цель: Различить по наблюдениям, что сделала модель, а что — приложение; показать, где хранится история разговора в чате и при обращении через API, и почему «лучшая модель вообще» — неверная постановка вопроса.

Формат: показ, а не практикум. Показывает учитель или видео автора; ученик до каждого показа записывает прогноз, после — наблюдение. Регистрация, установка и оплата от ученика не требуются; starter не используется.

Ход урока: 5 мин — сцена, вопрос-прогноз, карта «что есть что» в тетради; 7 мин — показ 1 «Куда мы попали?»; 9 мин — показ 2 «Что внутри»; 10 мин — показ 3 «Модели разные»; 9 мин — показ 4 «Второй вход: API»; 5 мин — таблица «чат / API», выходной билет. Итого 45 мин. Хронометраж — гипотеза до пилота.

Трек А: прогноз до каждого показа и таблица наблюдений (четыре строки: прогноз; что произошло; чья это работа — модели или приложения; вывод в одну фразу).

Трек В: необязательный — показы 1–3 повторить в чате, которым школа разрешает пользоваться, только вместе с учителем; результат — в ту же таблицу с пометкой «наш опыт». Личные данные не вводятся.

Распечатать или раздать: лист таблицы наблюдений; заготовка карты «что есть что» с двумя рамками-входами.

Проверка: мини-проверка «шесть наблюдений в два столбца» в итоге урока Э1 — ответы в книге сразу после неё; выходной билет: закончить фразу «В чате историю разговора хранит и присылает модели..., а при обращении через API — ...» (ожидается: приложение; программа, которую пишет автор). Владение словами первого знакомства (модель, приложение, контекст, версия модели, API, ключ API, обвязка) не проверяется.

Типичная ошибка: «ИИ — это сайт»; «модель сама помнит разговор и сама ищет в интернете».

Безопасность: личные данные, свои и чужие, в чат не вводятся; адрес сайта проверяется до ввода чего-либо; ключ API — секрет, его не показывают и не записывают в тетрадь.

Продукт и домашняя работа: четыре строки таблицы наблюдений и карта с двумя входами; домашней работы нет.

Упрощение и усложнение: упростить — в показе 3 обсудить только итоговую таблицу сравнения; усложнить — ученик предлагает свою задачу и критерий для набора показа 3.

Схема на 18 часов: один урок-показ на обе части — показы 1, 4, 5, 7 (только три роли), 8 и карта; показы 2, 3, 6 — домашний просмотр видео или чтение. Либо вся Экскурсия — домашним просмотром; тогда первые 5 минут занятия 1 — сверка таблиц наблюдений.

Урок Э2. Агент и его упряжка

Страница в книге:

Цель: Сравнить три входа к модели — чат, агент для кода и программу через API — по трём вопросам (что система может изменить, кто подтверждает действие, где записаны шаги) и показать на схеме папки агента файл инструкций, файл с записанной процедурой и файл подключений.

Формат: показ, как в уроке Э1. Требуется таблица наблюдений урока Э1. Самый плотный урок Экскурсии — держать темп показов.

Ход урока: 3 мин — возврат к прогнозу сцены, карта «что есть что»; 10 мин — показ 5 «Чат советует — агент делает»; 7 мин — показ 6 «Папка агента»; 9 мин — показ 7 «Skill и MCP»; 9 мин — показ 8 «Всё вместе» и итоговая таблица; 3 мин — «Какие бывают агенты»; 4 мин — возврат к сцене, выходной билет, мост к главе 1. Итого 45 мин. Хронометраж — гипотеза до пилота.

Трек А: прогноз до каждого показа; таблица наблюдений дополняется строками показов 5–8; карта «что есть что» заполняется до конца.

Трек В: нет.

Распечатать или раздать: лист таблицы наблюдений (продолжение); карта из урока Э1; схема папки агента (три файла: инструкции проекта, записанная процедура, подключения).

Проверка: мини-проверка «пять ситуаций — выбери вход» с ответами в книге; выходной билет: «Назови три вопроса, которые нужно задать любому агенту, прежде чем пустить его в папку» (ожидается: что может изменить; кто подтверждает; где записаны шаги); «Проверь себя» главы — с ответами в книге, разбираются в классе или дома. Таблица «Обязательные проверки» (должен / не должен / ошибка / граница) в конце урока — опора для разбора спорных ответов.

Типичная ошибка: «агент — это более умная модель»; «раз агент спросил подтверждение, действие безопасно».

Безопасность: агент запускается в отдельной папке; человек читает, что именно подтверждает; режим без вопросов — не знак безопасности, скорее наоборот; учётные записи — только с разрешения взрослых и по правилам сервиса, обходить ограничения сервиса или школы нельзя.

Продукт и домашняя работа: восемь строк таблицы наблюдений и заполненная карта «что есть что»; домашняя работа необязательна — таблица «что есть что» для одной знакомой системы, без её использования.

Упрощение и усложнение: упростить — показ 6 только по схеме папки; усложнить — объяснить, почему чат с поиском тоже можно назвать агентом.

Связь с проектом: шага проекта нет; на схеме ученик видит папки `skills/` и `workspace/my_agent/`, которые откроет в главах 9 и 1.

Урок 1. Что скрывается за одним окном чата

Страница в книге:

Цель: Развести роли человека, интерфейса, приложения и модели и показать, что убедительная фраза ещё не доказывает выполненное действие.

Ход урока: 5 мин — ситуация «чат обещает прочитать файл»; 10 мин — разбор ролей; 23 мин — работа парами со схемой потока данных; 7 мин — взаимная проверка схем. Итого 45 мин.

Трек А: разложить карточки «пользователь», «интерфейс», «приложение», «модель», «ответ» и собрать цепочку передачи данных.

Трек В: запустить локальную заглушку: `python chat_stub.py`.

Распечатать или раздать: карточки ролей и лист потока данных.

Проверка: мини-проверка с ответами сразу после неё; выходной билет ; «Проверь себя» главы — без напечатанных ответов, их сверяют по уроку 2.

Типичная ошибка: приписывать модели открытие окна, чтение файлов и запуск кода.

Безопасность: не помещать в учебные запросы пароли и реальные персональные данные.

Продукт и домашняя работа: схема «человек → приложение → модель → приложение → человек»; домашней работы нет.

Упрощение и усложнение: упростить до трёх ролей; усложнить добавлением сервера и сетевого интерфейса.

Урок 2. Когда помощник начинает выбирать

Страница в книге:

Цель: Отличить текстовое обещание от выполненного действия и объяснить, когда приложение становится агентом.

Ход урока: 4 мин — сравнение обещания и реального чтения; 11 мин — демонстрация; 23 мин — классификация «ответ / действие / агентный шаг»; 7 мин — разбор спорных случаев. Итого 45 мин.

Трек А: роли «модель», «приложение» и tool чтения карточки; участники передают запрос и наблюдаемый результат.

Трек В: сравнить обычный и ошибочный сценарии: `python chat_stub.py`, `python chat_stub.py --scenario missing_file`, затем машинно-читаемую форму `python chat_stub.py --json`.

Распечатать или раздать: три синтетические карточки задач и лист классификации.

Проверка: «Проверь себя на ходу» ; выходной билет урока 2 ; «Проверь себя» главы — без напечатанных ответов, их сверяют по тексту урока 2.

Типичная ошибка: считать агентом любую программу с чат-интерфейсом.

Безопасность: первые tools работают только на чтение.

Продукт и домашняя работа: цель проектного агента и три нецели; необязательно — разобрать знакомое ИИ-приложение по ролям.

Упрощение и усложнение: сравнить только чат и функцию; усложнить примером, где tool есть, а агентный цикл не нужен.

Урок 3. Один шаг и обратная связь

Страница в книге:

Цель: Восстановить агентный цикл и показать, как observation меняет следующий шаг.

Ход урока: 5 мин — два похожих файла; 8 мин — схема цикла; 25 мин — ролевая симуляция; 7 мин — восстановление цикла по памяти. Итого 45 мин.

Трек А: роли модели, приложения и двух tools; observations выдаются закрытыми карточками после действия.

Трек В: работать с loop_demo.py; выполнить `python loop_demo.py --scenario one_match` и `python loop_demo.py --scenario two_matches`.

Распечатать или раздать: карточки состояний, действий и observations.

Проверка: раздел урока и выходной билет; итоговая проверка главы —.

Типичная ошибка: считать observation пояснением, которое не влияет на продолжение работы.

Безопасность: действия симуляции обратимы и выполняются на копиях данных.

Продукт и домашняя работа: диаграмма agent loop проектного бюро; домашней работы нет.

Упрощение и усложнение: один проход цикла; усложнение — два разных observation после одного действия.

Урок 4. Четыре способа завершить запуск

Страница в книге:

Цель: Развести success, needs_input, error и limit и связать остановку с observation, retry и пределом шагов.

Ход урока: 5 мин — повтор одной ошибки; 10 мин — завершение, повтор и отказ; 20 мин — дополнение диаграмм; 3 мин — короткий показ учителя; 7 мин — контрольная точка. Итого 45 мин.

Трек А: карточки «повторить», «попросить данные», «остановиться», «передать человеку» и выбор корректного терминала.

Трек В: разобрать полный loop_demo.py; проверить пять сценариев: `python loop_demo.py --scenario one_match`, `python loop_demo.py --scenario two_matches`, `python loop_demo.py --scenario temporary_error`, `python loop_demo.py --scenario permanent_error`, `python loop_demo.py --scenario limit`; короткий учительский показ — `python agent_demo.py --scenario success`.

Распечатать или раздать: неполные блок-схемы и ошибочная карточка задачи.

Проверка: «Проверь себя на ходу»; выходной билет урока 4 с ответом для сверки; итог главы —.

Контроль: КТ 1: цикл — четыре терминала, пять сценариев, error на действии 2.

Типичная ошибка: «агент должен повторять, пока не получится».

Безопасность: предел шагов и повторов не даёт ошибке бесконечно расходовать ресурсы.

Продукт и домашняя работа: политика остановки: не более пяти шагов и одной повторной попытки чтения; добровольно — сравнить пять сценариев.

Упрощение и усложнение: только success и одна ошибка; усложнение — различить временную и постоянную ошибку.

Урок 5. Кто организует знакомый цикл

Страница в книге:

Цель: Показать harness как программную среду вокруг модели и распределить его обязанности.

Ход урока: 5 мин — «двигатель без машины»; 10 мин — схема компонентов; 23 мин — групповая сборка архитектуры; 7 мин — защита одной связи. Итого 45 мин.

Трек А: распределить карточки по группам «решение», «выполнение», «состояние», «контроль», «интерфейс».

Трек В: исследовать структуру `loop_demo.py`, `school_agent/`, `workspace/`, `tests/`, `workspace/traces/`; отдельного командного запуска в практике урока нет.

Распечатать или раздать: карточки компонентов и большая схема harness.

Проверка: мини-проверка ; выходной билет ; ответы главы — .

Типичная ошибка: считать harness разновидностью нейросети.

Безопасность: ограничения и permissions проектируются как часть harness, а не как пожелание в prompt.

Продукт и домашняя работа: архитектурная карта проектного агента v0.1; домашней работы нет.

Упрощение и усложнение: пять групп компонентов; усложнение — retries и управление контекстом.

Урок 6. Что видно сейчас и что действительно разрешено

Страница в книге:

Цель: Развести активный контекст, состояние и техническую границу доступа.

Ход урока: 4 мин — текстовый запрет против фактического запрета; 11 мин — демонстрация двух границ; 23 мин — сортировка данных и проектирование workspace; 7 мин — проверка схем. Итого 45 мин.

Трек А: распределить примеры по зонам «видит сейчас / хранится между запусками / недоступно».

Трек В: проверить доступ командами `python access_demo.py --access read --path tasks/task-003.json`, `python access_demo.py --access write --path tasks/task-003.json`, `python access_demo.py --access read --path tasks/missing.json`, `python access_demo.py --access read --path ../outside.json`; затем карту частей — `python harness_demo.py --show-parts`.

Распечатать или раздать: карточки данных и дерево каталогов.

Проверка: мини-проверка ; выходной билет ; ответы главы — .

Типичная ошибка: считать историю чата единственным состоянием.

Безопасность: скрытый в инструкции пароль не становится безопасным; права проверяются до операции.

Продукт и домашняя работа: политика рабочей папки и карта доступов; домашней работы нет.

Упрощение и усложнение: контекст против файла; усложнение — read-only и read-write.

Урок 7. Почему пять похожих слов обозначают разные механизмы

Страница в книге:

Цель: Различить prompt, structured output, tool, MCP и Agent Skill по роли, а не по слову в запросе.

Ход урока: 5 мин — пять решений одной задачи; 12 мин — таблица различий; 21 мин — парная классификация; 7 мин — пограничные случаи. Итого 45 мин.

Трек А: классифицировать карточки ситуаций и объяснить главный вопрос каждого механизма.

Трек В: показать фрагменты `python mechanisms_demo.py fragments`, затем проверить свои ответы `python mechanisms_demo.py check workspace/drafts/ch04_choices.txt`; для границ использовать готовые случаи из книги.

Распечатать или раздать: карточки ситуаций и сравнительную таблицу пяти механизмов.

Проверка: мини-проверка ; выходной билет ; ответы главы — .

Типичная ошибка: называть любой prompt или функцию Agent Skill.

Безопасность: протокол подключения не является доказательством доверия.

Продукт и домашняя работа: матрица пяти механизмов; домашней работы нет.

Упрощение и усложнение: оставить prompt, tool и Agent Skill; усложнить дополнительным разбором соседних механизмов.

Урок 8. Как механизмы работают вместе

Страница в книге:

Цель: Собрать host/client/server/resource и progressive disclosure в одну архитектуру без ложной лестницы «tool → MCP → Agent Skill».

Ход урока: 5 мин — каталог и вложенные материалы; 10 мин — «описание → SKILL.md → reference»; 23 мин — архитектурное ревью; 7 мин — рефлексия. Итого 45 мин.

Трек А: три вложенных конверта, моделирующие progressive disclosure.

Трек В: `python mechanisms_demo.py mcp`; затем `python mechanisms_demo.py skill --layer catalog`, `python mechanisms_demo.py skill --layer instructions`, `python mechanisms_demo.py skill --layer reference`.

Распечатать или раздать: три конверта или сложенных листа и карта архитектуры.

Проверка: мини-проверка ; выходной билет ; ответы главы — .

Контроль: КТ 2: архитектура — роли механизмов и связанная карта системы.

Типичная ошибка: изображать tool, MCP и Agent Skill как поколения, заменяющие друг друга.

Безопасность: внешний server и исполняемый script считаются недоверенными до проверки.

Продукт и домашняя работа: полная архитектурная схема проекта; домашней работы нет.

Упрощение и усложнение: без деталей client/server; усложнение — показать место согласия пользователя.

Урок 9. От просьбы к договору задачи

Страница в книге:

Цель: Зафиксировать цель, входы, результат, ограничения, неизвестность и критерий готовности до выбора tools и кода.

Ход урока: 5 мин — расплывчатая просьба; 10 мин — разбор хорошего договора; 23 мин — заполнение паспорта; 7 мин — чек-лист. Итого 45 мин.

Трек А: заполнить бумажный технический паспорт проекта.

Трек В: создать `task_spec.json` по структуре урока; отдельной команды запуска в практике нет.

Распечатать или раздать: шаблон договора задачи, плохие и хорошие примеры.

Проверка: мини-проверка ; выходной билет ; ответы главы — .

Типичная ошибка: описывать действие «обработать данные» вместо наблюдаемого результата.

Безопасность: запрещённые действия записываются в договоре заранее.

Продукт и домашняя работа: технический паспорт проекта v0.1; домашней работы нет.

Упрощение и усложнение: цель, вход и результат; усложнение — неизвестность и условия остановки.

Урок 10. Когда агент не нужен

Страница в книге:

Цель: Выбрать между обычным кодом, моделью и агентным циклом и не добавлять агента без необходимости.

Ход урока: 5 мин — три способа проверить поля; 10 мин — критерии выбора; 23 мин — классификация 12 задач; 7 мин — обсуждение разногласий. Итого 45 мин.

Трек А: таблица «обычный код / модель / агент» для двенадцати ситуаций.

Трек В: записать сигнатуры функций и входные значения без реализации; отдельного командного запуска в практике нет.

Распечатать или раздать: карточки задач и шаблон декомпозиции.

Проверка: мини-проверка ; выходной билет ; ответы главы — .

Типичная ошибка: добавлять agent loop только потому, что задача связана с ИИ.

Безопасность: чем выше цена ошибки, тем меньше оснований передавать решение автономной системе.

Продукт и домашняя работа: карта четырёх tools и список задач, которые агент не выполняет; домашней работы нет.

Упрощение и усложнение: выбрать из готовых вариантов; усложнение — сравнить две архитектуры по риску и сложности.

Урок 11. Как модель запрашивает tool

Страница в книге:

Цель: Восстановить цепочку tool calling и назвать исполнителя каждого шага.

Ход урока: 5 мин — имя и arguments ещё не действие; 10 мин — семишаговая схема; 23 мин — ролевая или программная симуляция; 7 мин — проверка цепочки. Итого 45 мин.

Трек А: модель пишет карточку вызова, приложение её проверяет, tool выполняет операцию и возвращает результат.

Трек В: `python tool_call_demo.py show`, затем `python tool_call_demo.py execute`.

Распечатать или раздать: карточки tool calls и схема цепочки.

Проверка: мини-проверка ; выходной билет ; ответы главы — .

Типичная ошибка: считать появление tool call доказательством выполнения.

Безопасность: запрос модели не является разрешением на выполнение.

Продукт и домашняя работа: схема первого tool проекта; домашней работы нет.

Упрощение и усложнение: один tool без аргументов; усложнение — два последовательных вызова.

Урок 12. Параметры как договор

Страница в книге:

Цель: Сделать arguments проверяемыми через schema и согласовать их с сигнатурой функции.

Ход урока: 5 мин — несовпадающее имя поля; 10 мин — элементы schema; 23 мин — исправление плохих описаний; 7 мин — короткий тест. Итого 45 мин.

Трек А: собрать schema из карточек полей и проверить правильный, неполный и недопустимый объект.

Трек В: `python tool_call_demo.py schema read_task_cards`, `python tool_call_demo.py schema validate_task_card`, `python tool_call_demo.py schema task_card`; для проверки — `python tool_call_demo.py check workspace/ch06/cases/validate_missing_task_id.json` и `python tool_call_demo.py check workspace/ch06/cases/read_with_path.json`.

Распечатать или раздать: карточки типов, полей и обязательности; шаблон schema.

Проверка: мини-проверка ; выходной билет ; ответы главы — .

Типичная ошибка: включить поле в `required`, но не описать его в `properties`.

Безопасность: arguments проверяются до вызова функции.

Продукт и домашняя работа: схема карточки задачи; домашняя работа — один дополнительный отрицательный пример.

Упрощение и усложнение: только строки и обязательные поля; усложнение — массив blockers и enum статусов.

Урок 13. Диспетчер и белый список

Страница в книге:

Цель: Связать имя tool только с разрешённой функцией через dispatch table и allowlist.

Ход урока: 5 мин — запрос `delete_everything`; 10 мин — таблица соответствий; 23 мин — сборка маршрутизатора; 7 мин — тест неизвестного имени. Итого 45 мин.

Трек А: диспетчер получает карточку запроса и передаёт её только разрешённому исполнителю.

Трек В: проверить три случая: `python dispatcher_demo.py case allowed`, `python dispatcher_demo.py case unknown-tool`, `python dispatcher_demo.py case invalid-arguments`.

Распечатать или раздать: карточки имён tool и таблицу allowlist.

Проверка: мини-проверка ; выходной билет ; ответы главы — .

Типичная ошибка: искать функцию по строке и вызывать её без белого списка.

Безопасность: не использовать `eval`, произвольные команды и подбор «похожего» имени.

Продукт и домашняя работа: диспетчер разрешённых tools; домашней работы нет.

Упрощение и усложнение: два tools; усложнение — метаданные режима доступа.

Урок 14. Ошибки, несколько вызовов и возврат результата

Страница в книге:

Цель: Обработать ошибки и несколько tool calls так, чтобы каждый результат становился наблюдаемым событием.

Ход урока: 5 мин — повреждённый файл; 10 мин — ветви обработки; 8 мин — офлайн-проверка ошибок; 15 мин — bridge и разбор; 7 мин — демонстрация результата. Итого 45 мин.

Трек А: карточки «успех / ошибка / неизвестный tool» и решение, что записать в журнал.

Трек В: `python dispatcher_demo.py case tool-exception`, `python dispatcher_demo.py run two-calls`, `python dispatcher_demo.py run stop-after-decision`, `python dispatcher_demo.py run missing-file`; локальная ошибка — `python agent_demo.py --scenario missing_file`. Для обязательного живого сеанса книга даёт команду `python agent_demo.py --adapter bridge --scenario success --trace traces/bridge-lesson14-group-a.jsonl`; порядок bridge —, паспорт —.

Распечатать или раздать: сценарии ошибок и карточки исходов; для bridge — шаблон паспорта.

Проверка: мини-проверка ; выходной билет ; ответы главы — .

Контроль: КТ 3: цикл и bridge — чтение, ошибка и результат; в треке В — паспорт bridge-сеанса.

Типичная ошибка: перехватить ошибку и не оставить о ней наблюдаемой записи.

Безопасность: текст ошибки не раскрывает секреты и лишние системные пути.

Продукт и домашняя работа: первый работающий локальный цикл; добровольно — разобрать уже выполненный bridge-сеанс.

Упрощение и усложнение: один тип ошибки; усложнение — несколько независимых результатов.

Урок 15. Хороший tool начинается с границы

Страница в книге:

Цель: Ограничить scope tool и сформулировать описание «что / над чем / когда / когда не применять».

Ход урока: 5 мин — review плохого helper; 10 мин — четыре правила описания; 23 мин — ревью tools проекта; 7 мин — обмен запросами. Итого 45 мин.

Трек А: заполнить карточку описания tool и проверить границы на двух запросах.

Трек В: `python tool_quality_demo.py passport validate_task_card`; затем `python tool_quality_demo.py route --case should-validate-card` и `python tool_quality_demo.py route --case should-read-board`.

Распечатать или раздать: плохие descriptions и паспорт tool.

Проверка: мини-проверка ; выходной билет ; ответы главы — .

Типичная ошибка: описание «работает с данными и помогает пользователю».

Безопасность: слишком широкий scope увеличивает число ложных вызовов и поверхность риска.

Продукт и домашняя работа: паспорта tools; домашней работы нет.

Упрощение и усложнение: один tool; усложнение — пересечение двух descriptions.

Урок 16. Validator и три обязательных теста

Страница в книге:

Цель: Проверить tool случаями «должен / не должен / ошибка» и отличить форму от замысла.

Ход урока: 5 мин — один успешный тест недостаточен; 10 мин — минимальный набор; 23 мин — практический зачёт; 7 мин — взаимное ревью. Итого 45 мин.

Трек А: провести три сценария с ролевым агентом.

Трек В: `python tool_call_demo.py check workspace/ch06/cases/validate_missing_task_id.json`, `python tool_quality_demo.py suite`, `python access_demo.py --access read --path ../outside.json`.

Распечатать или раздать: три карточки сценариев и чек-лист проверки.

Проверка: мини-проверка ; выходной билет ; ответы главы — .

Контроль: Проверочная работа 1 — tool, schema, validator и три теста.

Типичная ошибка: тестировать только «счастливым путем».

Безопасность: обязательный отрицательный случай — попытка обратиться к пути вне workspace.

Продукт и домашняя работа: проверенный пакет tools и отчёт тестирования; домашней работы нет.

Упрощение и усложнение: ручной чек-лист; усложнение — автоматический отчёт.

Урок 17. Что такое настоящий Agent Skill

Страница в книге:

Цель: Развести Agent Skill, prompt, tool и учебный каталог и определить роль SKILL.md.

Ход урока: 5 мин — три папки; 10 мин — структура каталога; 23 мин — проектирование skill; 7 мин — проверка обязательных элементов. Итого 45 мин.

Трек А: собрать бумажный каталог из карточек-файлов.

Трек В: `python skill_demo.py catalog`, `python skill_demo.py load project-status`, затем `python skill_demo.py check workspace/ch09/cases/project-status-valid` и `python skill_demo.py check workspace/ch09/cases/function-only`.

Распечатать или раздать: карточки файлов, шаблон паспорта skill и примеры каталогов.

Проверка: мини-проверка ; выходной билет ; ответы главы — .

Типичная ошибка: считать любые дополнительные папки обязательными для Agent Skill.

Безопасность: до подключения проверяются происхождение, scripts, сеть и permissions.

Продукт и домашняя работа: паспорт Agent Skill v0.1; домашней работы нет.

Упрощение и усложнение: только SKILL.md и reference; усложнение — совместимость и лицензия.

Урок 18. Пишем SKILL.md

Страница в книге:

Цель: Описать повторяемую процедуру компактно, с trigger, шагами, запретами, неизвестностью и критерием готовности.

Ход урока: 5 мин — слишком общий SKILL.md; 10 мин — структура процедуры; 15 мин — проектный skill; 8 мин — личный skill и три проверки; 7 мин — активационный тест. Итого 45 мин.

Трек А: бумажный SKILL.md и отдельные reference-карточки.

Трек В: `python skill_demo.py resource project-status references/report-template.md`; затем случаи `python skill_demo.py check workspace/ch09/cases/bad-frontmatter`, `python skill_demo.py check workspace/ch09/cases/script-present`, `python skill_demo.py check workspace/drafts/ch09/project-status` и `python skill_demo.py replay`.

Распечатать или раздать: шаблон SKILL.md, примеры процедур и чек-лист.

Проверка: мини-проверка ; выходной билет ; ответы главы — .

Контроль: КТ 4: skill — проектный и личный skill, три проверки.

Типичная ошибка: превращать SKILL.md в энциклопедию вместо порядка работы.

Безопасность: не хранить в skill реальные пароли, токены и персональные сведения.

Продукт и домашняя работа: проектный Agent Skill и личный учебный skill v1; добровольно — улучшить reference и проверки.

Упрощение и усложнение: шесть шагов процедуры; усложнение — детерминированный script после аудита.

Урок 19. Состояние, память, RAG и журнал — не одно и то же

Страница в книге:

Цель: Выбрать подходящее место для процедуры, текущего состояния, фактов и истории событий.

Ход

урока: 5 мин — где хранить «задача заблокирована»; 10 мин — карта механизмов; 23 мин — классификация 15 примеров; 7 мин — разбор. Итого 45 мин.

Трек А: разложить карточки по хранилищам.

Трек В: `python skill_demo.py load project-status` и `python state_demo.py classify`; затем отделить `project_state.json` от SKILL.md.

Распечатать или раздать: карточки данных и карта механизмов хранения.

Проверка: мини-проверка ; выходной билет ; ответы главы — .

Типичная ошибка: складывать все данные внутрь Agent Skill.

Безопасность: профиль и чувствительные сведения требуют отдельного решения и в проекте не используются.

Продукт и домашняя работа: модель состояния проекта; домашней работы нет.

Упрощение и усложнение: procedure/state/journal; усложнение — добавить RAG и профиль.

Урок 20. Как передать работу следующему запуску

Страница в книге:

Цель: Сохранить минимальное проверяемое состояние для продолжения работы без полной истории чата.

Ход урока: 5 мин — длинная стенограмма вместо состояния; 10 мин — файлы и защищаемые данные; 23 мин — передача проекта; 7 мин — проверка восстановления. Итого 45 мин.

Трек А: одна группа оставляет состояние, другая продолжает без устного объяснения.

Трек В: `python state_demo.py resume workspace/ch10/cases/project_state_ready.json`, `python state_demo.py resume workspace/ch10/cases/project_state_missing_next_step.json`, затем `python state_demo.py checkpoint workspace/ch10/cases/project_state_ready.json --out drafts/ch10/project_state.json`; проверить handoff командами из урока.

Распечатать или раздать: карточки хранилищ и шаблон handoff.

Проверка: мини-проверка ; выходной билет ; ответы главы — .

Контроль: КТ 5: состояние — handoff между запусками.

Типичная ошибка: сохранять всю переписку вместо проверяемого рабочего результата.

Безопасность: в handoff не включаются секреты и реальные персональные данные.

Продукт и домашняя работа: пакет возобновляемого состояния; домашней работы нет.

Упрощение и усложнение: один JSON-файл; усложнение — безопасная атомарная запись.

Урок 21. От теста к Eval

Страница в книге:

Цель: Превратить отдельные тесты в проверяемый eval-набор и зафиксировать baseline.

Ход урока: 5 мин — два красивых, но непроверенных отчёта; 10 мин — критерии и повторяемость; 23 мин — набор случаев; 7 мин — ревью критериев. Итого 45 мин.

Трек А: таблица «вход → ожидаемое действие → ожидаемый результат».

Трек В: `python eval_runner.py --summary`, затем `python eval_runner.py`; результаты harness и `model_behavior` рассматривать отдельно.

Распечатать или раздать: шаблон Eval case и примеры отчётов.

Проверка: мини-проверка ; выходной билет ; ответы главы — .

Типичная ошибка: оценивать ответ по впечатлению «умный и подробный».

Безопасность: в eval-набор включается запрос на запрещённое действие.

Продукт и домашняя работа: план eval-набора из шести случаев; дома — один граничный случай.

Упрощение и усложнение: три случая; усложнение — отдельные критерии маршрута и результата.

Урок 22. Сравниваем две версии на одном наборе

Страница в книге:

Цель: Сравнить версии на одинаковых входах и критериях и отделить реальное улучшение от удачного запуска.

Ход урока: 5 мин — версия В лучше на одном примере; 10 мин — одинаковые условия; 23 мин — слепое оценивание; 7 мин — вывод. Итого 45 мин.

Трек А: оценить два набора результатов, не зная названия версии.

Трек В: `python eval_runner.py --summary`; для проекта — `python dezhurny.py eval --dir workspace/my_agent/examples/full`.

Распечатать или раздать: два комплекта результатов и рубрику.

Проверка: мини-проверка ; выходной билет ; ответы главы — .

Контроль: Проверочная работа 2 — harness отдельно от `model_behavior`; baseline.

Типичная ошибка: менять входные данные одновременно с процедурой.

Безопасность: высокий общий балл не отменяет критического нарушения безопасности.

Продукт и домашняя работа: baseline-отчёт версии 1.0.0; домашней работы нет.

Упрощение и усложнение: pass/fail; усложнение — взвешенная рубрика.

Урок 23. Trace: что происходило на самом деле

Страница в книге:

Цель: Восстановить запуск по наблюдаемым событиям и спроектировать trace без скрытых рассуждений.

Ход урока: 5 мин — правильный итог неправильным маршрутом; 10 мин — структура trace; 23 мин — восстановление событий; 7 мин — проверка. Итого 45 мин.

Трек А: собрать trace из перемешанных карточек событий.

Трек В: `python dispatcher_demo.py run two-calls`; затем `python trace_demo.py record --scenario missing_file --out traces/ch12_lesson23.jsonl` и `python trace_demo.py show traces/ch12_lesson23.jsonl`.

Распечатать или раздать: карточки событий и шаблон JSONL.

Проверка: мини-проверка ; выходной билет ; ответы главы — .

Типичная ошибка: логировать только финальный текст.

Безопасность: trace хранит наблюдаемые события, а не пароли, чувствительные данные или скрытые рассуждения модели.

Продукт и домашняя работа: схема trace проекта; домашней работы нет.

Упрощение и усложнение: пять типов событий; усложнение — временные метки и call id.

Урок 24. Ищем ошибку по Trace

Страница в книге:

Цель: Найти первую failure point и отличить ошибку выбора, arguments, выполнения и обработки результата.

Ход урока: 5 мин — три traces с одинаковым плохим итогом; 8 мин — виды отказов; 25 мин — диагностика; 7 мин — защита вывода. Итого 45 мин.

Трек А: заполнить incident report по бумажному trace.

Трек В: `python trace_demo.py analyze workspace/ch12/cases/wrong_tool.jsonl`, `python trace_demo.py analyze workspace/ch12/cases/invalid_arguments.jsonl`, `python trace_demo.py analyze workspace/ch12/cases/tool_exception.jsonl`; в проекте — `python dezhurny.py run` и `python dezhurny.py trace --last`.

Распечатать или раздать: три намеренно повреждённых trace и бланк incident report.

Проверка: мини-проверка ; выходной билет ; ответы главы — .

Контроль: КТ 6: диагностика — incident report по trace.

Типичная ошибка: объяснять всё «ошибкой модели».

Безопасность: журнал должен позволять расследование, но не становиться хранилищем лишних данных.

Продукт и домашняя работа: incident report и локальное предложение исправления; домашней работы нет.

Упрощение и усложнение: одна явная ошибка; усложнение — несколько вторичных симптомов.

Урок 25. Как поправка превращается в опыт

Страница в книге:

Цель: Отличить разовую поправку от переносимого изменения процедуры, связанного с неуспешным Eval case.

Ход урока: 5 мин — повтор пропуска blocker; 10 мин — «ошибка → урок → предложение»; 23 мин — работа с примерами; 7 мин — ревью. Итого 45 мин.

Трек А: превратить три пользовательские поправки в правила или обоснованно отклонить их.

Трек В: `python trace_demo.py analyze workspace/ch12/cases/invalid_arguments.jsonl`; затем `python change_demo.py proposal workspace/ch13/change_proposals/validate-task-id.json` и `python eval_runner.py --summary`.

Распечатать или раздать: карточки feedback и шаблон change proposal.

Проверка: мини-проверка ; выходной билет ; ответы главы — .

Типичная ошибка: записывать в skill единичный факт вместо общего правила.

Безопасность: агент может предложить изменение, но не назначает его новой нормой сам.

Продукт и домашняя работа: change proposal личного skill v1→v2, связанный с одним неуспешным кейсом; домашней работы нет.

Упрощение и усложнение: один пример; усложнение — проверка конфликта нового правила со старым.

Урок 26. Version, Regression и Rollback

Страница в книге:

Цель: Сравнить Version по diff и прежним тестам и принять keep/revert без потери старой версии.

Ход урока: 5 мин — одна ошибка исправлена, две появились; 10 мин — цикл изменения; 23 мин — сравнение версий; 7 мин — решение. Итого 45 мин.

Трек А: две распечатанные Version и четыре одинаковые карточки Test case; отметить diff, повторить случаи и принять keep/revert.

Трек В: `python change_demo.py diff workspace/ch13/versions/1.0.0/project-status workspace/ch13/versions/1.1.0-candidate/project-status`; затем `python change_demo.py compare workspace/ch13/versions/1.0.0/project-status workspace/ch13/versions/1.1.0-candidate/project-status --cases workspace/ch13/version_cases.json` и хороший вариант из книги.

Распечатать или раздать: обязательно две короткие распечатки одной процедуры и четыре одинаковые карточки Test case; шаблон release note.

Проверка: мини-проверка ; выходной билет ; ответы главы — .

Контроль: КТ 7: версии — личный skill v1/v2, diff, повтор случаев, keep/revert.

Типичная ошибка: удалить старую Version до проверки новой.

Безопасность: возможность Rollback — часть безопасного улучшения.

Продукт и домашняя работа: Version 1.1.0 либо обоснованный Rollback; домашней работы нет.

Упрощение и усложнение: ручное сравнение; усложнение — Git commit и автоматическая регрессия.

Урок 27. Модель угроз и минимальные полномочия

Страница в книге:

Цель: Связать каждое разрешение с конкретной операцией и применить least privilege.

Ход урока: 5 мин — чтению дали право удаления; 10 мин — простая модель угроз; 23 мин — аудит архитектуры; 7 мин — защита ограничения. Итого 45 мин.

Трек А: таблица «операция → необходимый доступ → запрещённый доступ».

Трек В: проверить техническую границу записи на знакомом примере `python access_demo.py --access write --path tasks/task-003.json` и сопоставить её с матрицей permissions урока.

Распечатать или раздать: карточки угроз, дерево файлов и шаблон permissions.

Проверка: мини-проверка ; выходной билет ; ответы главы — .

Типичная ошибка: выдавать максимальные права «чтобы точно работало».

Безопасность: технические ограничения сильнее словесного запрета.

Продукт и домашняя работа: модель угроз и матрица разрешений; домашней работы нет.

Упрощение и усложнение: чтение/запись; усложнение — сеть, процессы и временные разрешения.

Урок 28. Человек внутри цикла

Страница в книге:

Цель: Спроектировать approval gate до значимого действия и развести предложение системы и решение человека.

Ход урока: 5 мин — изменение статуса без разрешения; 10 мин — уровни автономности; 23 мин — подтверждение; 7 мин — практический зачёт. Итого 45 мин.

Трек А: агент формирует предложение, человек проверяет diff и принимает решение.

Трек В: `python permissions_demo.py policy --dir workspace/my_agent`; затем `python permissions_demo.py approval --decision deny`, `python permissions_demo.py approval --decision approve` и граничные случаи из урока.

Распечатать или раздать: карточки действий и шаблон экрана подтверждения.

Проверка: мини-проверка ; выходной билет ; ответы главы — .

Контроль: Проверочная работа 3 — permissions и approval gate.

Типичная ошибка: спрашивать подтверждение после выполнения действия.

Безопасность: в проекте нет автоматического удаления, публикации и внешней отправки.

Продукт и домашняя работа: работающий approval gate; домашней работы нет.

Упрощение и усложнение: решение да/нет; усложнение — показать diff и причину действия.

Урок 29. Prompt injection внутри обычного документа

Страница в книге:

Цель: Распознать недоверенный текст как данные и ограничить его последствия техническими границами.

Ход урока: 5 мин — команда внутри карточки; 10 мин — разбор атаки; 23 мин — заражённые документы; 7 мин — обсуждение защиты. Итого 45 мин.

Трек А: маркировать фрагменты как «данные», «инструкция системы», «подозрительный текст».

Трек В: `python tool_call_demo.py execute`, затем `python trust_demo.py document --case task-901` и `python trust_demo.py replay`.

Распечатать или раздать: синтетические документы с инъекциями и лист анализа.

Проверка: мини-проверка ; выходной билет ; ответы главы — .

Типичная ошибка: искать только явную фразу «игнорируй предыдущие инструкции».

Безопасность: текстовый фильтр недостаточен; последствия ограничиваются permissions и approval gate.

Продукт и домашняя работа: протокол обработки недоверенного содержимого; домашней работы нет.

Упрощение и усложнение: явная инъекция; усложнение — скрытая команда в таблице или reference.

Урок 30. Аудит чужого skill и MCP-подключения

Страница в книге:

Цель: Проверить provenance, scripts, зависимости, сеть и permission scope до подключения внешнего skill или MCP-стороны.

Ход урока: 5 мин — полезный skill просит широкие права; 10 мин — чек-лист доверия; 23 мин — командный аудит; 7 мин — защита решения. Итого 45 мин.

Трек А: аудит muck-пакетов и карточки MCP-сервера по чек-листу.

Трек В: `python trust_demo.py audit skill --sample project-status`, `python trust_demo.py audit skill --sample project-sync`, `python trust_demo.py audit skill --sample suspicious-skill`, `python trust_demo.py audit mcp --sample mock-mcp-server`.

Распечатать или раздать: обязательно распечатки `SKILL.md`, дерева файлов и текста scripts пакетов `project-status` и `project-sync`, а также карточку `mock_mcp_server.json`.

Проверка: мини-проверка ; выходной билет ; ответы главы — .

Контроль: КТ 8: аудит доверия — внешний аудит skill другой группы с обоснованным отказом или условиями подключения.

Типичная ошибка: «открытый стандарт означает, что источнику можно доверять».

Безопасность: подключение начинается с минимального доступа; решение `isolate` значит «не подключать до дополнительной проверки» — отдельной защищённой среды (sandbox) starter не создаёт.

Продукт и домашняя работа: отчёт аудита и решение о подключении; домашней работы нет.

Упрощение и усложнение: готовый чек-лист; усложнение — лицензия, совместимость и транзитивные зависимости.

Урок 31. Один проект — несколько запусков

Страница в книге:

Цель: Восстановить проект после завершения Session по resumable state и не повторить уже выполненное действие.

Ход урока: 5 мин — работа прервана после третьего шага; 10 мин — inner/outer loop; 23 мин — передача проекта; 7 мин — восстановление. Итого 45 мин.

Трек А: одна группа оставляет допустимый handoff, вторая продолжает по нему.

Трек В: `python state_demo.py resume workspace/ch10/cases/project_state_ready.json`; затем `python dezhurny.py status --chapter 16`, несколько запусков `python dezhurny.py run --dir workspace/ch16/cases/long-work` и граничная проверка `python long_work_demo.py repeat`.

Распечатать или раздать: неполное состояние и checklist восстановления.

Проверка: мини-проверка ; выходной билет ; ответы главы — .

Типичная ошибка: сохранять только сообщение «мы почти закончили».

Безопасность: возобновление не должно повторно применять уже подтверждённое изменение.

Продукт и домашняя работа: инструкция восстановления после сбоя; домашней работы нет.

Упрощение и усложнение: один checkpoint; усложнение — operation id и защита от повторного выполнения.

Урок 32. Один агент, несколько агентов или человек

Страница в книге:

Цель: Выбрать одноагентную, многоагентную или человеческую схему и назвать владельца итогового решения.

Ход урока: 5 мин — два агента меняют одно состояние; 10 мин — критерии разделения; 23 мин — анализ архитектур и утверждений; 7 мин — решение. Итого 45 мин.

Трек А: сравнить «один агент», «два агента», «агент + человек».

Трек В: `python long_work_demo.py handoff --case analyzer`, затем отрицательный случай `python long_work_demo.py handoff --case missing-owner`.

Распечатать или раздать: карточки архитектур и кейсы для обсуждения.

Проверка: мини-проверка ; выходной билет ; ответы главы — .

Типичная ошибка: считать multi-agent архитектуру автоматически более умной.

Безопасность: агент не принимает кадровые, медицинские, юридические или финансовые решения.

Продукт и домашняя работа: записка об архитектурном выборе и границах применения; домашней работы нет.

Упрощение и усложнение: один агент или человек; усложнение — конфликт состояния и протокол handoff.

Урок 33. Интеграция: проект как единая система

Страница в книге:

Цель: Выполнить согласованный полный прогон tools, skill, состояния, evals и ограничений и зафиксировать Release candidate.

Ход урока: 4 мин — критерии готовности; 8 мин — демонстрация полного прогона; 26 мин — командная интеграция; 7 мин — обмен проектами. Итого 45 мин.

Трек А: ролевая симуляция полного маршрута по карточкам и бумажному trace.

Трек В: `python dezurny.py candidate --label festival-rc1 --dir workspace/ch17/cases/release-candidate`, `python dezurny.py status --chapter 17 --dir workspace/ch17/cases/release-candidate`, `python dezurny.py run --dir workspace/ch17/cases/release-candidate`, `python dezurny.py eval --dir workspace/ch17/cases/release-candidate`, затем `python eval_runner.py --summary`.

Распечатать или раздать: финальный checklist, рубрику защиты и набор синтетических данных.

Проверка: мини-проверка ; выходной билет ; ответы главы — .

Контроль: Интеграция — кандидат проекта для итоговой проверки .

Типичная ошибка: улучшать презентацию при непройденном критическом тесте.

Безопасность: перед обменом удаляются временные данные; permissions фиксируются и не расширяются.

Продукт и домашняя работа: Release candidate, постер архитектуры, папка доказательств; дома — репетиция защиты.

Упрощение и усложнение: защита только трека А; усложнение — дополнительный API-adapter, не влияющий на основную версию.

Урок 34. Публичная защита и инженерная рефлексия

Страница в книге:

Цель: Доказать понятность, проверяемость и ограниченность системы через проект, неуспешный сценарий и безопасную реакцию.

Ход урока: 5 мин — подготовка рабочих мест; 28 мин — защиты; 7 мин — взаимное оценивание; 5 мин — индивидуальная рефлексия. Итого 45 мин.

Трек А: показать ролевую модель, документы, trace и eval-таблицу.

Трек В: показать локальный запуск подготовленного Release candidate и свидетельство минимум одного bridge-сеанса; новых обязательных команд в уроке 34 книга не вводит, используются проверенные артефакты урока 33.

Распечатать или раздать: рубрику защиты, форму самооценки и форму взаимного ревью.

Проверка: мини-проверка ; выходной билет ; рубрика защиты и ответы главы .

Контроль: Итог — защита, bridge-доказательство трека В и личный skill; рубрика.

Типичная ошибка: демонстрировать только успешный красивый сценарий и скрывать ограничения.

Безопасность: критическое нарушение не компенсируется презентацией; обязательны минимальные permissions и решение человека там, где оно требуется.

Продукт и домашняя работа: публичная защита, личный skill с тремя проверками и решением keep/revert, индивидуальная рефлексия; домашней работы нет.

Упрощение и усложнение: упрощение — защита минимального обязательного маршрута; усложнение — доказанное улучшение и переносимость по бонусной части рубрики.