AI Coding 规范：从自由探索到按规矩办事

课程逐字稿（讲师讲解词）

第四章

共 38 页幻灯片

第 1 页

【幻灯片标题】AICoding 规范：从自由探索到按规矩办事

【逐字稿】

同学们好，欢迎来到第四章《AI Coding 规范：从自由探索到按规矩办事》。在第三章中，我们学习了 Vibe Coding 的核心流程和实战技巧，体验了用自然语言指挥 AI 写代码的快感。但大家在实际操作中可能已经发现了一个问题——Vibe Coding 虽然快，但生成的代码质量不可控，缺乏统一规范，在团队协作和正式项目中会出大问题。这一章我们就来解决这个问题，学习 SDD 规格驱动开发体系，掌握 AI 标准化编程思维。如果说 Vibe Coding 是「自由探索」，那 SDD 就是「按规矩办事」。两种模式不是对立的，而是互补的——先自由探索验证想法，再按规矩办事交付项目。让我们开始吧！

第 2 页

【幻灯片标题】Page1

【逐字稿】

本章主要讲述 AI Coding 规格驱动开发体系，颠覆传统编程模式，告别无规范的随性编码，聚焦 AI 标准化编程思维。课程主要讲解几个核心内容：第一，AI 编程差异——理解有规范和无规范的区别。第二，沟通技巧——如何与 AI 有效协作。第三，需求拆解与提示词设计——如何把模糊需求转化为精确规格。第四，代码审核——如何验收 AI 产出的代码。第五，AI 智能排错——如何让 AI 帮你找 bug 修 bug。通过这些内容，你将掌握 AI 规范编程的全闭环，依托 AI 工具独立完成项目开发、调试与交付工作。这一章实操性很强，我们会通过一个天气出行小助手的案例来贯穿全流程。

第 3 页

【幻灯片标题】Page2

【逐字稿】

我们来看本章的学习目标。学完本课程后，您将能够：第一，了解 SDD 规格驱动 AI 编程的核心价值与设计思想——知道为什么需要规范，规范带来什么价值。第二，掌握 AI 编程协作规范，包含标准化对话与结构化提示词设计——学会用规范的方式与 AI 沟通。第三，学习 AI 规范化开发全流程，涵盖需求拆解、代码审核与智能调优——掌握从需求到交付的完整链路。第四，实践项目 SDD 实战——写规格书改造天气出行小助手，动手体验 SDD 全流程。这四个目标从认知到技能到流程到实践，层层递进。本章的核心是实践——SDD 不是理论，是方法论，只有动手做了才能真正掌握。

第 4 页

【幻灯片标题】Page3

【逐字稿】

我们来看本章的目录。本章分为五个部分：第一部分是「SDD 规范的落地价值」，理解为什么需要规范。第二部分是「SDD 是什么：先写说明书，再让 AI 动工」，理解 SDD 的核心理念。第三部分是「学会写一份 AI 能懂的规格书」，掌握规格书的写作方法。第四部分是「从规格到代码的完整流程」，掌握 SDD 的四步开发流程。第五部分是「Vibe vs SDD：什么时候自由，什么时候规矩」，理解两种模式的选择策略。这五部分从价值到概念到方法到流程到选择，形成完整的 SDD 知识体系。我们逐一展开。

第 5 页

【幻灯片标题】Page4

【逐字稿】

我们先来回顾 Vibe Coding 的魅力。Vibe Coding 是 AI 时代的「手搓」快乐——一句话生成代码，无需繁琐的逐行编写，只需一句自然语言指令，AI 即可快速生成可用代码，将开发效率直接拉满。过程自由随性，打破传统编程的严谨束缚，想到哪写到哪，随时调整思路与方向，在探索式的交互中迸发更多意想不到的创造力。「Vibe Coding」一词由前特斯拉 AI 总监、OpenAI 联合创始人 Andrej Karpathy 在 2025 年 2 月首次提出。他描述这是一种全新的编码方式——你完全沉浸在氛围中，拥抱指数级的效率提升，甚至忘记代码本身的存在。Vibe Coding 确实很爽，但我们要清醒地认识到——爽不等于好，快不等于对。接下来我们看无规范的 Vibe Coding 在复杂场景下会出什么问题。

第 6 页

【幻灯片标题】Page5

【逐字稿】

我们来看无规范的 Vibe Coding 在复杂场景下的问题。Vibe Coding 的自由开发模式在轻量场景下能充分释放效率，但进入系统化、生产级开发场景时，问题就来了——仅凭个人理解向 AI 下达指令、缺乏统一标准与共识，会让 AI 生成的代码失去对齐基准，从逻辑设计到实现细节全面脱节。举个具体例子：同样围绕「用户登录认证」需求，三次不同的 Vibe Coding 产出完全不同：第一次「写个登录功能」——仅模糊描述业务诉求，生成代码偏基础实现，自由度高但适配性弱。第二次「实现用户认证接口」——侧重后端接口，未定义参数格式和返回值标准，接口与业务层衔接存在断层。第三次「做一个账号密码验

证模块」——聚焦单一功能，未考虑系统集成，复用性极低。这就是场景升级的必然结果——系统碎片化，质量全面失控。同一个需求，三次产出三种风格，无法整合，无法维护。这就是为什么我们需要 SDD。

第 7 页

【幻灯片标题】Page6

【逐字稿】

我们来看两类开发场景的核心差异，理解什么时候需要规范。场景 A 是轻量原型或独立开发——效率优先的灵活模式。代码标准是结果导向，灵活度高。以快速验证需求为核心目标，变量命名、注释、格式均可灵活调整，只要功能可运行、开发方可理解，无需额外规范约束。这种场景 Vibe Coding 是最佳选择。场景 B 是生产级系统建设——标准先行的质量模式。代码标准是长期维护，全域对齐。系统由多模块、多环节串联构成，统一的命名规范、注释标准、代码风格是基础要求。若各模块仍沿用随性开发模式，代码库将成为「风格大杂烩」，无论是模块联调、人员交接还是后续迭代，都会产生极高的理解与沟通成本。这种场景必须用 SDD。核心认知：不是所有场景都需要规范，但生产级项目必须规范。Vibe Coding 做原型，SDD 做交付。

第 8 页

【幻灯片标题】Page7

【逐字稿】

我们进入第二部分——SDD 是什么。前面讲了无规范的痛点和两类场景的差异，现在正式定义 SDD。这部分会讲清楚 SDD 的核心理念、规格标准化方法、以及三种开发模式的本质差异，帮助大家全面理解 SDD 的定位和价值。

第 9 页

【幻灯片标题】Page8

【逐字稿】

我们来正式定义 SDD——规格驱动开发。核心理念是「规格即真相，写好说明书，AI 照着办」。告别随意发挥的生成式开发，以明确、可执行的规格为核心，锚定 AI 产出的准确性与一致性。SDD 有三个核心角色：第一，开发者共创——通过详尽、无歧义的规格书，明确需求边界、功能细节与验收标准。第二，AI 严格执行——将规格书作为 AI 的唯一输入指令，确保代码输出严格匹配预设的业务规则与技术规范。第三，统一审查基准——所有代码审查和功能讨论均以规格书为基准，消除主观理解偏差，聚焦目标达成。协作共赢的结果是：统一 AI 输出标准，建立团队共同的沟通语言，大幅降低协作成本，让技术交付更高效、更精准，从源头减少返工与误解。SDD 的本质是把「人脑里的隐性知识」转化为「文档上的显性规格」，让 AI 和所有人都能精确理解和执行。

第 10 页

【幻灯片标题】Page9

【逐字稿】

我们来看规格标准化的具体方法。规格书分为四个模块：业务层——业务流程、正常流程、全量异常边界场景；接口层——入参出参格式、数据类型、取值边界、校验规则、

统一异常返回码；技术层——技术实现约束、代码规范、依赖组件、性能指标、安全要求；验收层——可量化、可自动化测试的判定标准。输出统一规范规格文档，作为人和 AI 唯一执行依据。AI 定向代码生成的原则是：以完整规格文档作为 AI 唯一输入指令，禁止碎片化、口语化零散需求。AI 严格依据规格内定义的参数、逻辑、集成约束生成代码，天然规避功能重叠、逻辑割裂、模块无法兼容问题，统一代码风格、数据标准、异常处理逻辑。这四个模块从业务到接口到技术到验收，层层细化，确保 AI 生成的代码在各个维度都符合要求。

第 11 页

【幻灯片标题】Page10

【逐字稿】

我们来看三种开发模式的本质差异。传统开发：完全依赖人工手动编码与调试，代码严谨规范、稳定性高，但开发周期长、效率偏低，适用于大型企业级团队和强合规要求场景。Vibe Coding：以 AI 为核心自由生成代码，开发速度极快能快速产出原型，但生成结果不可控易出现逻辑漏洞，适用于个人探索性项目和原型快速验证。SDD 开发：基于严格的规格定义引导 AI 按既定规则精准执行，既保留 AI 的生成速度，又通过规格约束确保代码质量与可维护性，适用于需要团队高效协作、代码需长期迭代维护的企业级应用开发。核心公式：SDD = Vibe Coding 的极致速度 + 传统开发的严谨规范。SDD 融合两者优势，实现开发效率与系统可控性的完美平衡，是 AI 时代更具落地价值的开发范式。理解这个公式，就理解了 SDD 的定位。

第 12 页

【幻灯片标题】Page11

【逐字稿】

我们进入第三部分——学会写一份 AI 能懂的规格书。SDD 的核心是规格书，规格书的质量直接决定代码质量。这部分会讲清楚规格书的核心要素、AI 能懂的规格书标准、结构化表达方法，并通过一个登录功能的完整规格书示例来演示。

第 13 页

【幻灯片标题】Page12

【逐字稿】

我们来看规格书的四大核心要素。一句名言：「一份严谨的规格书，是项目从需求到落地的桥梁。它不仅是开发团队的执行说明书，更是产品交付的质量保证书。」四个要素：第一，功能描述——核心聚焦「做什么」，明确产品或需求的核心目标与功能范畴，是规格书的基础锚点，确保研发与设计方向不偏离业务初衷。第二，输入输出——厘清「用什么、产出什么」，明确接口数据、资源输入条件及最终交付成果形式，界定系统交互边界。第三，约束条件——界定「有什么限制」，明确时间周期、预算成本、技术实现难点、环境依赖等限制因素，提前识别风险。第四，验收标准——回答「怎么算合格」，制定可量化、可验证的交付指标与质量规范，作为需求验收的核心依据。这四要素缺一不可——缺功能描述不知道做什么，缺输入输出不知道交互什么，缺约束条件不知道边界在哪，缺验收标准不知道是否合格。

第 14 页

【幻灯片标题】Page13

【逐字稿】

我们来看 AI 能懂的规格书标准。三个核心标准：第一，清晰——每句话只有唯一含义，避免模糊表述。第二，完整——包含所有正常流程与异常边界场景。第三，可验证——可转化为具体检验条件，客观判断结果。我们对比看一个例子。反面案例：「实现登录功能，要好用」——问题在于「好用」是主观抽象的感受，没有明确的判定标准，开发与测试无法对齐认知，最终交付结果会偏离预期。正面案例：「用户输入正确的账号密码后，系统应在 2 秒内返回包含 token 的 JSON 格式响应」——明确了触发条件、响应时效与数据格式，可直接通过技术手段验证是否达成。区别就在于：正面案例的每句话都可以用代码验证，反面案例的「好用」无法验证。写规格书时，每写完一句就问自己：这句话能写成测试用例吗？如果不能，就重写。

第 15 页

【幻灯片标题】Page14

【逐字稿】

我们来看结构化表达的方法。四个技巧：第一，用列表分点代替大段文字——避免冗长的段落叙述，将核心信息拆解为条目化内容，让阅读者能快速抓取关键要点。第二，用「如果…则…」描述分支逻辑——清晰界定条件与结果的对应关系，明确不同场景下的执行路径，消除模糊的逻辑表述。第三，明确数据类型、格式与范围——对每一个数据字段都定义具体类型（如字符串、数字）、格式规范（如日期格式）和取值范围，确保数据标准统一无歧义。第四，结合实际案例说明输入输出——通过真实的输入输出示例，让抽象的规则变得具象，帮助读者快速理解规格的实际应用场景。结构化表达的本质是消除歧

义——把可能被多种理解的表达转化为只有一种理解的表达。这对 AI 尤其重要——AI 不像人可以追问确认，它按照字面意思执行，所以规格书必须精确到只有一种理解。

第 16 页

【幻灯片标题】Page15

【逐字稿】

我们通过一个完整的登录功能规格书示例来演示。先看功能描述：用户在登录界面输入账号与密码后，系统将提交信息至后端服务进行鉴权验证。若验证通过，为用户生成合法的登录态 Token 并写入客户端，允许访问核心业务功能；若验证失败，精准返回对应错误提示信息。再看输入参数定义：username——数据类型字符串，长度 3-20 字符，格式仅支持英文字母、数字及下划线，禁止中文空格特殊符号，需全局唯一。password——数据类型字符串，长度 8-32 字符，必须同时包含大小写字母、数字及特殊符号，后端存储需不可逆加密。核心原则是输入校验是系统首道安全防线——规范参数边界、格式、异常逻辑，从源头规避安全及业务异常。注意每个字段都定义了类型、长度、格式、约束——这就是结构化表达，AI 看到这些就知道该怎么写校验代码。

第 17 页

【幻灯片标题】Page16

【逐字稿】

我们继续看登录功能规格书的安全约束和验收标准。核心约束条件包含三个安全规则：第一，传输加密——密码传输全程采用高强度加密算法，禁止明文传输，从链路层面杜绝信息泄露。第二，风控锁定——同一账号连续登录失败达 5 次后，系统自动触发 15

分钟锁定，抵御暴力破解。第三，会话时效——Token 有效期严格限定为 2 小时，超时自动失效需重新验证。三者结合形成安全闭环。验收标准包含四个场景：鉴权成功场景——输入正确账号密码，系统返回有效 Token 并建立会话。凭证错误场景——账号不存在或密码不匹配，返回「账号或密码错误」且不暴露具体细节。参数合规性——密码未达要求时即时拦截并引导修正。锁定机制验证——连续 5 次失败后账号锁定，期间登录被拒绝。注意验收标准都是可测试的——你可以直接写成测试用例来验证。这就是好的规格书——每个要求都可以被检验。

第 18 页

【幻灯片标题】Page17

【逐字稿】

我们进入第四部分——从规格到代码的完整流程。前面讲了怎么写规格书，现在讲怎么用规格书来驱动 AI 生成代码。SDD 开发分为四个阶段：提案、审查、实施、归档。这部分会逐一讲解每个阶段的工作流程和实操方法。

第 19 页

【幻灯片标题】Page18

【逐字稿】

我们来看 SDD 开发的四个步骤。第一步，提案阶段——编写规范规格书，明确业务、技术要求，使 AI 精准理解需求，锁定正确开发方向。第二步，审查阶段——生成代码后对照规格逐项校验功能、逻辑与边界场景，修正问题，达标交付。第三步，实施阶段——合格代码接入开发环境部署联调，保障新模块和现有系统兼容稳定。第四步，归档阶段—

—归档规格文档与验证代码，留存完整开发记录，支撑运维、排障与迭代沉淀。这四步形成一个闭环——提案定义做什么，审查验证做得对不对，实施把代码跑起来，归档留下记录供后续参考。注意顺序不能乱——先提案再审查再实施再归档。很多人容易跳过提案直接实施，这就是 Vibe Coding 的做法，在简单场景可以，但在复杂项目中会出问题。

第 20 页

【幻灯片标题】Page19

【逐字稿】

我们来看提案阶段的工作流。共四步：第一步，需求分析——与产品、业务方深度对齐核心需求，明确目标方向，全面梳理功能点覆盖范围，排查各类异常场景与边界条件。第二步，撰写规格书——紧扣功能、输入输出、约束、验收四大核心要素编写，采用结构化表达形式，确保文档逻辑清晰、内容完整、具备可验证可落地的实操性。第三步，团队评审——邀请技术、产品等团队成员联合审查规格书，重点校验内容是否存在遗漏、描述模糊或逻辑不合理的问题。第四步，规格书定稿——根据评审意见修订，确认无异议后正式确定最终版本，作为后续开发测试的核心依据。提案阶段是 SDD 中最重要的阶段——规格书写好了，后面的一切都顺理成章；规格书写不好，后面处处是坑。所以提案阶段不要急，多花时间把规格想清楚写清楚。

第 21 页

【幻灯片标题】Page20

【逐字稿】

我们来看工程文件的标准化命名。统一文件命名规则属于 SDD 底层标准化体系。三个核心文件：README.md 是项目总说明，SPEC.md 是业务规格书，CONTRIBUTING.md 是开发贡献规范。文件命名为通用开源行业约定，团队无需额外解释即可识别文件用途。SPEC.md 专用于存放完整需求规格，是 SDD 开发模式的核心基准文件。重点理解 SPEC.md 的定位——它不只是一份前期文档，而是贯穿开发全链路的唯一校验标准。全流程以该文件为标尺，统一需求口径、代码逻辑、边界处理规则。标准化文件名保障所有开发人员、AI 工具取用同一份规格基准。如果没有统一命名，极易出现多版本规格、信息割裂问题。所以 SDD 项目中一定要有一份 SPEC.md，所有人都以它为准。

第 22 页

【幻灯片标题】Page21

【逐字稿】

我们来看提案阶段的实操——编写新功能规格书。以「天气分享海报生成」功能为例，规格书包含六部分：功能描述——功能目的、用户场景、操作流程。输入输出——输入是天气数据（温度、天气状况、AQI、出行建议），输出是 PNG 格式海报图片，数据流向是天气数据→canvas 绘制→图片生成→下载分享。校验规则——P0 级：输入数据不完整时的处理、生成频率限制（30 秒内最多 3 次）；P1 级：图片尺寸要求（1080x1920）；P2 级：海报文案的 AI 生成规则。安全约束——canvas 禁止 innerHTML、不得包含用户注入内容、下载必须用户主动触发。UI/UX 规范——按钮位置样式、loading 状态、toast 提示文案。验收标准——3 秒内生成、内容与天气数据一致、成功失败都有提示。注意校验规则标注了优先级 P0/P1/P2——P0 必须实现，P1 应该实现，P2 可选实现。这种优先级标注帮助 AI 和开发者聚焦关键要求。

第 23 页

【幻灯片标题】Page22

【逐字稿】

我们来看审查阶段。审查是验证 AI 生成的代码是否完全符合规格书要求。这是确保最终交付成果准确、可用的核心环节，需要像专业质检员一样，拒绝「差不多」的敷衍式交付。审查四个维度：第一，功能是否落地——逐项核对需求，全覆盖功能逻辑，保障业务目标落地。第二，数据格式校验——校验接口入参、值域、返回结构，对齐上下游交互契约。第三，规则是否合规——核查性能、权限、业务规则，提前规避安全与逻辑隐患。第四，异常是否覆盖——覆盖极值、空值、超时等异常场景，验证系统容错稳定性。发现偏差时的关键原则：不手动修改生成的代码，而是回归问题本质——修正规格书的描述。将发现的问题明确、量化地补充进规格书中，再重新调用 AI 生成。这个原则保证了「规格书即真相」，让文档与代码始终保持高度一致，避免后续维护出现「两张皮」现象。

第 24 页

【幻灯片标题】Page23

【逐字稿】

我们来看审查阶段的工作流。核心原则是坚持「规格书是代码的唯一真相来源」，通过迭代自然语言描述而非手动修改代码，确保系统设计的一致性与可维护性。共四步循环：代码生成——导入定稿规格书，AI 自动生成初始代码。规格审查——对照规格书逐条核验代码实现。迭代规格——代码不达标时不直接改代码，而是优化规格补充约束。重新生成——迭代生成、核验、修正规格闭环，直至代码达标。这个循环很重要——不是一次

生成就完事，而是可能需要多轮迭代。每轮迭代都是修正规格书，而不是手动改代码。这样做的好处是：规格书始终是最新最全的，后续任何人拿到规格书就能理解系统。如果手动改代码而不更新规格书，规格书就和代码脱节了，变成「两张皮」。

第 25 页

【幻灯片标题】Page24

【逐字稿】

我们来看审查阶段的规格书评审实操。为什么要做规格书评审？一个角色看不到的问题，换个角色就能看到——这就是评审的价值。以技术评审委员会的身份评审「天气分享海报」功能规格书，评审要点有四个：第一，完整性——是否有遗漏的功能点？考虑无网络时能否生成、天气数据为空时怎么办、生成时间过长是否有取消按钮。第二，准确性——描述是否清晰无歧义？「海报风格统一」太模糊需要具体说明、「生成成功提示」要明确文案。第三，可行性——技术实现是否有障碍？canvas 在移动端性能如何、中文特殊字体是否需要加载。第四，一致性——是否与现有规格书冲突？颜色风格是否与主题一致、频率限制是否与现有功能冲突。评审后输出评审报告，包含通过项、不通过项、建议项和总体结论。根据评审意见修改规格书，再进入下一步。这个评审环节能发现单人思考的盲区，是质量保障的重要一环。

第 26 页

【幻灯片标题】Page25

【逐字稿】

我们来看实施阶段的工作流。严格遵循标准发布流程，做好环境配置与数据验证，实现代码平稳集成与部署。共四步：第一步，本地开发与测试——本地调试 AI 生成代码，编写单元测试校验全量逻辑与边界值，满足验收标准。第二步，创建特性分支——从主干拉取特性分支开发，隔离代码不影响线上稳定版本。第三步，提交与代码审查——分支提交后发起 MR，团队评审把控代码规范与质量。第四步，合并与自动部署——评审通过合并主干，触发 CI/CD 自动构建、测试、部署。实施阶段的关键是不要跳过测试——AI 生成的代码虽然通过了规格审查，但仍需要单元测试和集成测试来保证运行时正确性。同时，通过分支开发和 MR 评审，确保团队对代码质量有共识。

第 27 页

【幻灯片标题】Page26

【逐字稿】

我们来看实施阶段的 AI 生成代码和对照检查实操。将编写好的规格书作为输入，让 AI 严格按照规范重新生成代码。关键提示词要求：严格遵循规格书，不添加规格书未定义的功能；约束优先，必须遵守所有 P0 级校验规则和安全约束；规范命名，CSS 类名使用 BEM 规范；安全编码，禁止使用 innerHTML，APIKey 使用配置变量；完整健壮，包含 loading 状态和错误处理。AI 生成代码后，需要对照规格书逐条检查——每个功能点是否实现，每个校验规则是否遵守，每个安全约束是否满足。如果发现偏差，不手动改代码，而是修正规格书后重新生成。这个环节验证了规格书的有效性——如果 AI 严格按照规格书生成的代码能通过审查，说明规格书写得好；如果总是不达标，说明规格书有问题，需要优化。

第 28 页

【幻灯片标题】Page27

【逐字稿】

我们来看归档阶段的工作流。归档实现从代码实现到业务需求的双向追溯，打通需求、开发与维护的全链路信息闭环。共三步：第一步，绑定版本——规格书绑定 Git 提交 ID，代码变更可溯源对应需求，规范版本对应关系。第二步，记录变更日志——在文档或代码注释记录变更原因、内容、时间，留存完整变更记录，支撑维护审计。第三步，统一存储——规格书、源码、测试用例、评审材料统一归档仓库或 Wiki，搭建统一可检索知识库。归档的价值在于：后续维护人员无需反复研读复杂的代码细节，通过查阅归档的规格书即可快速理解设计初衷与业务逻辑，显著缩短问题定位与修复周期，提升迭代效率。这就是为什么 SDD 说「规格书即项目文档」——好的规格书本身就是最好的文档。

第 29 页

【幻灯片标题】Page28

【逐字稿】

我们来看归档阶段的需求变更处理。当需求变更时，遵循「先改规格书，后改代码」的原则。变更流程四步：提案——为新功能编写规格书章节；审查——评审新规格书；实施——基于新规格书生成代码；归档——更新主规格书，记录变更日志。以「天气分享海报」功能为例，完成开发后将该功能的规格书章节合并到主规格书 SPEC.md 中，更新版本号和修订历史，标注新增内容，更新功能模块列表。关键原则：版本精准绑定——将技术规格书与对应代码版本永久关联保存，确保技术文档与实际代码实现逻辑始终保持高度一致。变更全程留痕——详细记录修改人、执行时间及业务动因，形成完整可追溯的变更

历史档案。这个变更流程保证了规格书和代码始终同步——不会出现代码改了但文档没更新的情况。

第 30 页

【幻灯片标题】Page29

【逐字稿】

我们来看有约束和无约束的代码质量对比，直观感受 SDD 的价值。无约束的代码存在四个问题：风格混乱——代码格式命名混乱，可读性差维护成本高；校验缺失——缺少参数与边界校验，易出现运行异常；安全隐患——未过滤外部输入，存在 SQL 注入 XSS 等隐患；注释缺失——业务逻辑无注释，代码晦涩难迭代。有约束的代码有四个优势：风格统一——遵循编码规范，格式命名统一，代码清晰易读；校验完备——全链路参数校验异常捕获，系统运行更稳定；安全合规——输入清洗加数据脱敏，规避安全攻击风险；注释清晰——核心逻辑附带注释，降低理解成本提升协作效率。给 AI 设定规范、校验、安全约束，可产出生产级高质量代码，从源头杜绝劣质代码。这就是 SDD 的核心价值——不是增加负担，而是通过前期约束减少后期的质量问题。

第 31 页

【幻灯片标题】Page30

【逐字稿】

我们进入第五部分——Vibe vs SDD 的选择策略。前面讲了 SDD 的完整流程，现在回到一个实际问题：什么时候用 Vibe Coding，什么时候用 SDD？这部分会讲清楚两种模式的适用边界和最佳配合策略。

第 32 页

【幻灯片标题】Page31

【逐字稿】

我们来看 Vibe Coding 的适用场景。Vibe Coding 适合快速试错、验证想法。具体场景包括：探索阶段——需求未定，快速试错验证业务，暂缓架构规范设计。原型开发——极速产出 Demo，具象产品，缩短需求沟通周期。个人项目——灵活开发，舍弃协作规范，优先落地速度。核心优势是敏捷响应、轻量高效——开发效率高成本低，前期可省去繁杂流程与架构设计，快速落地创意，快速收集用户和市场反馈。潜在局限是维护与协作壁垒——缺少标准化架构让后期逻辑难以管控，迭代后维护成本大幅上涨；无统一开发规范、缺少完整文档，不支持多人协作。所以 Vibe Coding 只适用于单人短期小型项目，无法承载长期迭代的大型商业项目。记住：Vibe Coding 做原型可以，做产品不行。

第 33 页

【幻灯片标题】Page32

【逐字稿】

我们来看 Vibe 和 SDD 的最佳配合策略。最佳配合是：Vibe 探索→规格沉淀→SDD 交付→持续迭代。探索阶段用 Vibe 快速试错，低成本验证方案可行性。规格沉淀阶段把落地成果整理为规格书，固化统一开发标准。SDD 交付阶段依托规格生成规范代码，全流程管控交付质量。持续迭代阶段需求变更先改规格再迭代代码，实现前后统一。「快速探索+规范落地」双模式结合 Vibe 敏捷创新与 SDD 标准化开发，规避纯敏捷代码难维护、传统工程效率低的问题，兼顾灵活与规范，是新一代 AI 高效研发模式。这个配合策略的

核心理念是：不同阶段用不同模式——前期探索用 Vibe 求快，后期交付用 SDD 求稳。不是选一个用到底，而是根据阶段灵活切换。

第 34 页

【幻灯片标题】Page33

【逐字稿】

我们来看「双脑工作流」——左手 Vibe，右手 SDD。左脑 Vibe 自由探索——主打低成本快速试错，不受流程限制，快速产出多套方案验证思路，迭代筛选最优创新方向，充分发挥 AI 创作能力。右脑 SDD 规范落地——以规范设计为核心，方案敲定后搭建严谨架构、统一代码标准、留存完整文档，打造易维护扩展的系统，有序落地创意，保障业务长期稳定运行。双脑协同——Vibe 快速验证创意方向，SDD 标准化保障交付质量，二者互补形成闭环，兼顾开发速度与落地规范。落地策略是灵活切换、动态适配——无需固定单一模式，可按项目阶段、团队能力灵活切换，平衡创新灵活度与工程稳定性，最大化研发效率。这个双脑比喻很形象——人有两个脑半球各司其职，AI 编程也有两种模式各有所长。灵活切换，才是最高效的工作方式。

第 35 页

【幻灯片标题】Page34

【逐字稿】

我们来做一个本章小结。第一，梳理了从 Vibe 编码到 SDD 标准化开发的思维转变逻辑——从自由探索到按规矩办事。第二，掌握了规格文书写作规范，熟练搭建规格四大核心内容——功能描述、输入输出、约束条件、验收标准。第三，熟知 SDD 四段开发流程

——提案、审查、实施、归档，依托规格校验 AI 产出代码。第四，明晰 Vibe 与 SDD 适用边界——Vibe 做原型 SDD 做交付，按需组合两种开发模式。第五，掌握项目落地思路——平衡开发效率与工程规范性，双脑工作流灵活切换。这五点构成了 SDD 的完整知识体系。核心认知：SDD 不是增加负担，而是通过前期规范减少后期返工——磨刀不误砍柴工。

第 36 页

【幻灯片标题】Page35

【逐字稿】

我们来看一道思考题。SDD 标准化开发四阶段正确流程为？选项 A：提案→审查→实施→归档——这是正确的，先提案写规格，再审查验代码，然后实施部署，最后归档留记录。选项 B：提案→实施→审查→归档——顺序错了，审查应该在实施之前，先审查代码是否合格再实施部署。选项 C：审查→提案→实施→归档——顺序错了，提案应该在最前面，先有规格才能审查。选项 D：提案→审查→归档→实施——顺序错了，归档应该在最后，实施完成后才归档。正确答案是 A。这道题帮助大家记住 SDD 的四步顺序——提案定义规格，审查验证代码，实施部署上线，归档留存记录。顺序不能乱。

第 37 页

【幻灯片标题】Page36

【逐字稿】

给大家一些学习推荐资源。华为云官网是 https://www.huaweicloud.com/，可以了解华为云的 AI 编程产品和服务。华为云学堂是 https://edu.huaweicloud.com/，有丰富的 AI 编程

课程和实验环境。建议大家课后在码道平台上动手实践 SDD 全流程——写一份规格书，让 AI 按规格生成代码，审查验收，体验从规格到代码的完整链路。SDD 是方法论，只有动手做了才能真正理解。好，本章内容就到这里，感谢大家的聆听！