

Qoder使用指南

产品概述

Qoder (/ˈkɒdər/) 是一款面向真实软件开发的 Agentic 编码平台。通过增强上下文工程与智能体无缝结合，全面理解你的代码库，并以系统化方式推进开发任务。它提供代码智能生成、智能问答、多文件修改、编程智能体等能力，思考更深入、编码更高效、构建更出色，为开发者带来高效、流畅的编码体验。

快速开始

1 下载并安装 Qoder

- 从**Qoder**官网下载安装程序。
- 双击文件开始安装。
- 双击 Qoder IDE 图标启动 Qoder。

2 登录

- 在 Qoder IDE 右上角，点击用户图标，或使用键盘快捷键（  , (macOS) 或   , (Windows) ），然后选择 登录。
- 在打开的网页中：
 - 点击底部的 注册 并完成注册
 - 或使用 Google 或 GitHub 账号直接注册。
- 返回 Qoder IDE。你现在可以自由使用所有功能。

3 打开项目

可选择使用本地项目，或从 GitHub 克隆示例项目。

- 使用本地项目
 - 点击 打开 或使用键盘快捷键：
 - macOS:  
 - Windows:  
 - 浏览到项目文件夹，选择文件并打开。
- 克隆代码库
 - 点击 Clone repo。
 - 在顶部搜索栏中：

- 输入项目 URL 并点击 Clone from URL，或
- 点击 Clone from GitHub 并按提示操作。
- 按步骤完成项目克隆。

4 探索功能

触发 NEXT

NEXT 会在光标处提供智能、基于上下文的编辑，帮助你快速开始 AI 辅助编码。

- 输入部分代码片段或用自然语言描述代码需求。例如：“初始化一个list。”
- 按 `⌘ P` (macOS) 或 `Alt P` (Windows)，建议会自动出现。
- 按 Tab 接受建议。

NES 支持多行编辑与流畅的自动补全。

```
2 import { sunoApi } from "@lib/SunoApi":
3 export const dynamic = "force-dynamic";
4 export async function POST (req: NextRequest) {
5   if (req.method === 'POST') {
6     try {
7       const body = await req.json();
8       const { prompt } = body;
9
10      if (!prompt) {
11        return new NextResponse(JSON.stringify({ error: 'Prompt is required' })), {
12          status: 400,
13          headers: { 'Content-Type': 'application/json', ...corsHeaders
14        } );
15      }
16
17      const lyrics = await (await sunoApi).generateLyrics(prompt);
18
19      return new NextResponse(JSON.stringify(lyrics), {
20        status: 200,
21        headers: { 'Content-Type': 'application/json'
22      } );
23    } catch (error:any) {
24      console.error('Error generating lyrics:', JSON.stringify(error.response.data));
25      if (error.response.status === 402) {
26        return new NextResponse(JSON.stringify({ error: error.response.data.detail })), {
27          status: 402,
28          headers: { 'Content-Type': 'application/json',
29        } );
30      }
31    }
32  }
```

发起行间对话

使用行间会话在代码的上下文中直接获得 AI 帮助。

在 Qoder 代码编辑器中，按 `⌘ I` (macOS) 或 `Ctrl I` (Windows)。行间会话窗口将打开。

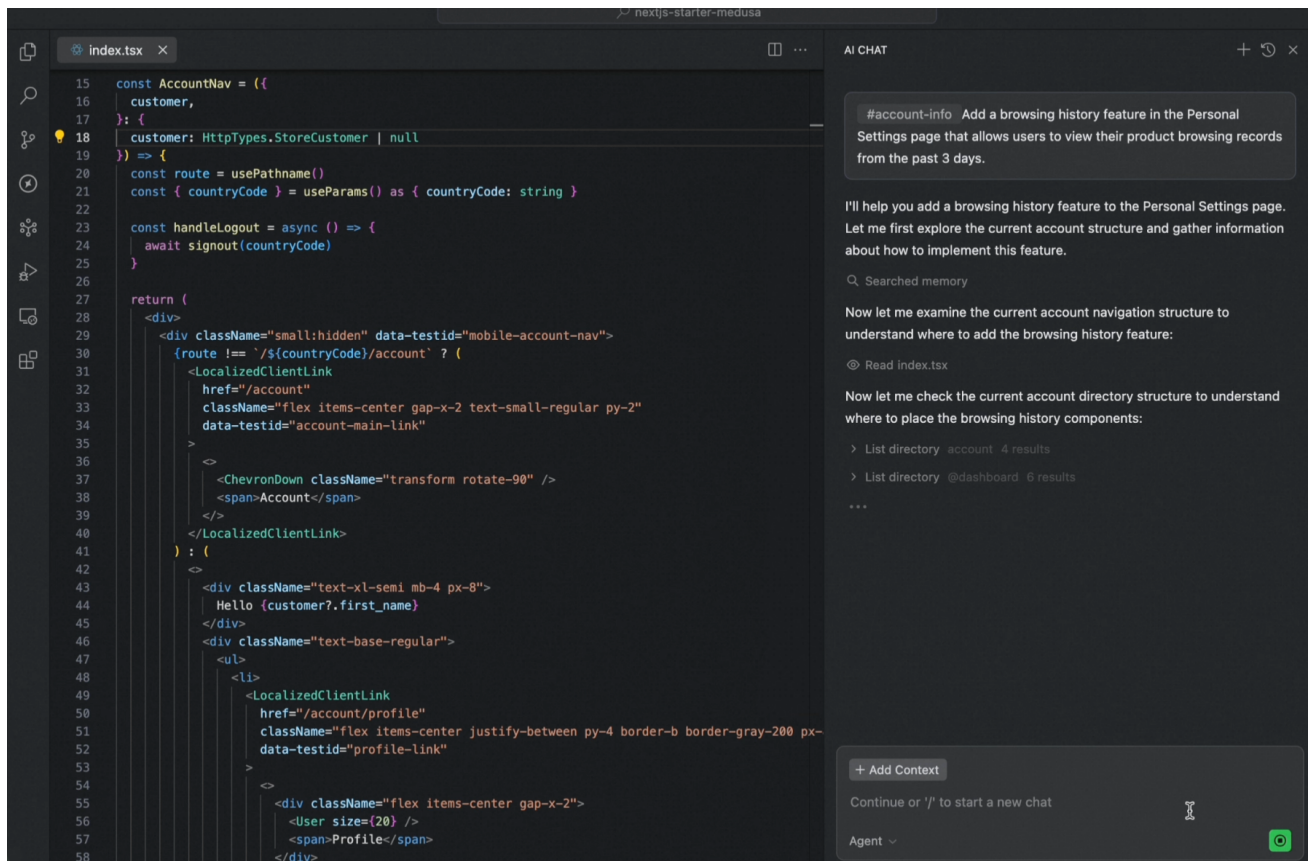
输入请求并发送。例如：“新增一个文件上传处理函数”

应用 AI 生成的代码，按 `⌘ ↵` (macOS) 或 `Ctrl Enter` (Windows)。

开始 AI 对话

使用智能会话面板，或按 `⌘ L` (macOS) 或 `Ctrl L` (Windows)，在智能问答或智能体模式下处理更复杂的任务。

- 在右侧的智能会话面板中输入请求。例如：“为这个函数创建并运行测试”
- 发送请求。智能体将生成包含相关用例的测试文件。
- 按提示点击 执行，或按 `⌘ ↵` (macOS) 或 `Ctrl Enter` (Windows) 执行测试。



RepoWiki

分析你的代码并生成结构化文档。它将隐性知识显性化。

REPO WIKI

> Component Architecture

> GENERATION STATUS

> CATALOG

Project Overview

Technology Stack & Depend...

Getting Started

> Routing & Navigation

> Data Fetching & State Manag...

> API Integration Layer

The nextjs-starter-medusa project is a frontend storefront implementation designed to work seamlessly with Medusa, a headless commerce platform. It leverages modern web development technologies to deliver a high-performance, scalable, and maintainable e-commerce experience.

The core technology stack includes:

- **Next.js 15:** Utilizing the App Router, React Server Components, Server Actions, and advanced caching mechanisms for optimal performance and SEO.
- **TypeScript:** Providing type safety and improved developer experience.
- **Tailwind CSS:** Enabling utility-first styling for rapid UI development.
- **Medusa JS SDK:** Facilitating communication between the frontend & the Medusa backend via RESTful APIs.

This architecture follows a modular component design, separating concerns into distinct directories such as modules, lib, and components. The use of React Server Components allows for efficient data fetching directly on the server, reducing client-side JavaScript payload and improving load times.

mermaid

Frontend (Next.js)

App Router

React Server Components

Medusa JS SDK

Medusa Backend API

Tailwind CSS

UI Rendering

Server Actions

State Management

Quest 自主 Agent 能力

一个能够明确意图并完成端到端任务的智能体，无需持续的人工指导。

Create an intense arcade-style space shooter game with a stunning neon aesthetic on a pure black background..

Thought 7s

I'm refining the "Neon Space Shooter – Cyberpunk Edition" concept using EnterQuestModeTool, focusing first on scope, visuals (starting with a pure black background), and a solid foundation: the game loop and canvas setup.

Execution Suggestion Spec first ✓

Thought 3s

Create to-dos ↻

Spec Written Plan to write a spec according to your reply

Thought 7s

neon-space-shooter-cyberpunk.md +45

My initial spec file is complete and ready to go.. I'm transitioning from the planning phase.

Spec neon-space-shooter-cyberpunk.md

I will transform the existing foundation into a high-polish cyberpunk space shooter with advanced visual effects (bloom, chromatic aberration, glitch), synthetic audio generation, holographic UI, and responsive mobile controls.

View detail Build

@ for context, / for commands

Local

neon-space-shooter-cyberpunk.md

Neon Space Shooter - Spec

Overview

This plan outlines the steps to enhance the existing neon-space-shooter.html into a polished, high-performance cyberpunk arcade shooter with advanced visual effects, synthetic audio, and responsive controls.

Refactoring & Foundation

- Refactor monolithic script into modular classes (Game, Renderer, Input, AudioManager, EntityManager).
- Implement a robust Entity base class for all game objects.
- Set up a responsive canvas system that handles device pixel ratio and dynamic resizing (4:3 aspect ratio).

Advanced Visual Effects

- Bloom Effect: Use an off-screen canvas for a 2-pass Gaussian blur and composite it for a neon glow.
- Chromatic Aberration: Implement RGB channel offsets triggered by damage and special events.
- Glitch Effects: Add scanline offsets and block displacements during impacts and phase changes.
- Warp Speed: Animate starfield acceleration and radial blur during transitions or "Time Slow" power-up.
- Enhanced Trails: Improve motion trails with gradients and varying alpha for all moving objects.

Web Audio Synthesis

- AudioManager: Initialize Web Audio API.
- Sound Synthesis:
 - Laser: Sine sweep (high frequency).
 - Explosion: Low sine + noise burst for deep bass.
 - Power-up: Arpeggio sine waves.
- Synthwave Music: Create an 8-bar loop using oscillators (bass, lead, pad).

UI & HUD Enhancements

- Holographic HUD: Add scanlines and glowing holographic borders to health/shield bars.
- Neon Grid Background: Implement a 3D perspective grid effect for the Game Over screen.
- Animated Menus: Add pulsing glow and sound feedback to buttons and titles.
- Combo Multiplier: Add dynamic scaling and pulsing animations for the combo counter.

4 / 4