Códigos demoscene, terminal, CMD, PowerShell e DOSBox

Uma lista brutalmente prática de repositórios e códigos para aprender editando e rodando.

65 links curados - jogos, animações ASCII, shaders, ferramentas de demo, retrocoding DOS e engines de terminal.

Como usar este PDF

- Comece pelos itens 1 a 6. Eles dão vocabulário e evitam que você vire colecionador de repo sem produzir nada.

- Depois escolha um trilho: PowerShell/CMD para resultado rápido, DOSBox para baixo nível, ou shaders para demoscene moderna.

- Clone um repo por vez, rode como está, mude uma variável visual, depois reescreva um efeito pequeno do zero.

- Não execute binários aleatórios fora de sandbox. Para demos antigas, prefira DOSBox, VM ou código compilado por você.

Atalho honesto

Se você tentar aprender tudo ao mesmo tempo, vai aprender nada. Faça esta ordem: 1) PowerShell Matrix, 2) clippy/ascii-animator, 3) ASCII Donut, 4) Bonzomatic + Book of Shaders, 5) dos-dev-template + Tetris Assembly, 6) Second Reality só depois.

Roteiro de aprendizado recomendado

Fase 1 - Terminal básico

CursorPosition, cores ANSI, limpar tela, loop de tempo, input sem bloquear. Projetos: Matrix.ps1, clippy, ascii-animator.

Fase 2 - Matemática visual

Seno, cosseno, rotação 2D/3D, projeção, z-buffer simples. Projetos: ASCII Donut, AsciiCube, ascii-3d-cube.

Fase 3 - Jogos de console

Loop de jogo, colisão, estado, score, teclado, sprites texto. Projetos: dotnet-console-games, CSharpConsoleGames, tetrigo.

Fase 4 - DOSBox e baixo nível

Modo 13h, memória de vídeo, interrupções BIOS, Assembly 8086, C/DJGPP. Projetos: masm-tasm, dos-dev-template, dos3d, Tetris Assembly.

Fase 5 - Shader/demoscene moderna

GLSL, fragment shader, SDF, raymarching, sincronia com música. Projetos: Book of Shaders, Bonzomatic, SDF, Rocket.

Fase 6 - Produção

Monte uma demo curta de 30 a 60 segundos: intro, 3 efeitos, música ou beep/track, créditos, empacote e grave vídeo.

Comandos-base

PowerShell:

```
git clone URL_DO_REPO cd pasta Get-ChildItem .\script.ps1
```

CMD/Node/Python:

```
git clone URL_DO_REPO cd pasta python main.py node index.js
```

DOSBox:

```
mount C C:\dosprojetos C: cd PROJETO programa.exe
```

Lista curada de códigos e repositórios

Cada item abaixo tem um caminho de execução sugerido e o que você deve tentar aprender. Links em azul são clicáveis.

0 - Comece por aqui

01. Teach Yourself Demoscene in 14 Days

Roda em: Leitura + projetos guiados Linguagem: Markdown / referencias Nível: Iniciante

Descrição: Guia direto para entrar na demoscene sem ficar perdido em ferramenta. Use como trilha principal antes de mergulhar nos códigos mais duros.

Aprenda: vocabulário da cena, categorias de demo, papéis de coder/artist/musician, pipeline mental

Abrir código/repositório
`https://github.com/psenough/teach_yourself_demoscene_in_14_days`

02. Demoscene Starter Kits

Roda em: Windows, Linux, Processing, plataformas variadas Linguagem: Processing / exemplos Nível: Iniciante

Descrição: Repositório criado para dar pontos de partida práticos. É ótimo para parar de só assistir demos e começar a alterar alguma coisa.

Aprenda: boilerplate, loop visual, primeiros efeitos, estrutura mínima de demo

Abrir código/repositório
`https://github.com/anttihirvonen/demoscene-starter-kits`

03. Awesome Demoscene

Roda em: Mapa de estudo Linguagem: Lista curada Nível: Todos

Descrição: Índice grande de recursos da demoscene. Não é para rodar, é para garimpar. Use como lista-mãe.

Aprenda: onde encontrar engines, produções, ferramentas, fontes e comunidades

Abrir código/repositório
`https://github.com/psykon/awesome-demoscene`

04. in4k.github.io

Roda em: Browser / documentação Linguagem: Markdown / HTML Nível: Intermediário

Descrição: Material clássico para entender intros pequenas. A mentalidade aqui é: cada byte tem que justificar sua existência.

Aprenda: sizecoding, 1k/4k intros, truques para reduzir tamanho

Abrir código/repositório
`https://github.com/in4k/in4k.github.io`

05. 64k Scene

Roda em: Browser / documentação Linguagem: HTML / JS Nível: Intermediário

Descrição: Galeria e documentação de 64k intros. Útil para ver o padrão de qualidade e entender o que estudar depois.

Aprenda: 64k intros, geração procedural, síntese de áudio, composição audiovisual

Abrir código/repositório
`https://github.com/64k-scene/64k-scene.github.io`

06. The Book of Shaders

Roda em: Browser / editor GLSL Linguagem: GLSL / JS Nível: Iniciante a avançado

Descrição: A ponte mais limpa entre matemática visual e demoscene moderna. Rode no navegador, edite valores, quebre tudo e volte.

Aprenda: fragment shaders, ruído, formas, padrões, raymarching inicial

Abrir código/repositório
`https://github.com/patriciogonzalezvivo/thebookofshaders`

1 - Demoscene real e intros com fonte

07. Second Reality - Future Crew

Roda em: DOSBox para binário; código histórico para estudo Linguagem: Assembly / C / Pascal Nível: Avançado

Descrição: Código-fonte de uma das demos mais importantes do MS-DOS. Não comece por ela, mas volte quando quiser estudar demoscene raiz de verdade.

Aprenda: VGA, efeitos 2D/3D, sincronização com música, organização modular de demo grande

Abrir código/repositório
`https://github.com/mtuomi/SecondReality`

08. Oscar's Chair

Roda em: Windows / build C++ Linguagem: C++ / GLSL Nível: Avançado

Descrição: Intro 4k vencedora em evento grande. Boa para estudar como uma cena rica nasce de pouco código e muita matemática.

Aprenda: 4k intro, raymarching, compactação mental de cena

Abrir código/repositório
`https://github.com/demoscene-source-archive/oscar-s-chair`

09. Terrarium

Roda em: Windows / build C++ Linguagem: C++ / shader Nível: Avançado

Descrição: Outro estudo forte de intro pequena. O foco é aprender como o visual nasce sem assets tradicionais.

Aprenda: natureza procedural, composição, textura gerada por código

Abrir código/repositório
`https://github.com/demoscene-source-archive/terrarium`

10. Ohanami

Roda em: Windows / build C++ Linguagem: C++ / shader Nível: Avançado

Descrição: Bom para comparar decisões artísticas com restrição brutal de tamanho.

Aprenda: estética procedural, animação compacta, shader minimalista

Abrir código/repositório
`https://github.com/demoscene-source-archive/ohanami`

11. Love Reaction

Roda em: Windows / build C++ Linguagem: C++ / GLSL Nível: Avançado

Descrição: Intro pequena com fonte. Use para estudar estrutura de uma produção completa, não apenas um efeito solto.

Aprenda: transições, cenas geradas, compactação de recursos

Abrir código/repositório
`https://github.com/demoscene-source-archive/love-reaction`

12. Horizon Machine

Roda em: Windows / build C++ Linguagem: C++ / shader Nível: Avançado

Descrição: Útil para entender como vender escala e atmosfera com pouco código.

Aprenda: paisagens, distância, iluminação procedural, narrativa visual curta

Abrir código/repositório
`https://github.com/demoscene-source-archive/horizon-machine`

13. Glittermorphosis

Roda em: Windows / build C++ Linguagem: C++ / shader Nível: Avançado

Descrição: Boa referência para efeitos de transformação e brilho dentro do vocabulário de intros.

Aprenda: morphing, partículas, estilo visual compacto

Abrir código/repositório
`https://github.com/demoscene-source-archive/glittermorphosis`

14. Dropletia

Roda em: Windows / build C++ Linguagem: C++ / shader Nível: Avançado

Descrição: Se você quer estudar efeito líquido sem engine enorme, este é o tipo de fonte que importa.

Aprenda: água, materiais, superfície procedural

Abrir código/repositório
`https://github.com/demoscene-source-archive/dropletia`

15. Alive Here Now, Forever

Roda em: Windows / build C++ Linguagem: C++ / GLSL Nível: Avançado

Descrição: Fonte de intro para estudar estética e engenharia juntas.

Aprenda: composição audiovisual curta, sincronização, proceduralismo

Abrir código/repositório
`https://github.com/demoscene-source-archive/alive-here-now-forever`

16. Appear - Jetlag

Roda em: Windows / build C++ Linguagem: C++ / shader Nível: Avançado

Descrição: Boa peça para estudar estilo mais moderno de 4k/intro.

Aprenda: shader intro, estrutura de build, cena comprimida

Abrir código/repositório
`https://github.com/w23/jetlag_appear`

17. Crawlspace

Roda em: Windows / build conforme repo Linguagem: Assembly / C / shader Nível: Avançado

Descrição: Material para quando você quiser sofrer de forma produtiva. 1k intro é treino de faca afiada.

Aprenda: 1k intro, obsessão por bytes, truques mínimos

Abrir código/repositório
`https://github.com/in4k/crawlspace`

18. Star Traveler

Roda em: Browser / local server / Node opcional Linguagem: JavaScript Nível: Intermediário

Descrição: Ótima ponte entre demoscene e JavaScript: mais fácil de editar que C++ com toolchain antiga.

Aprenda: demo pequena em JS, canvas/WebGL, animação compacta

Abrir código/repositório
`https://github.com/depp/demo-traveler`

2 - Ferramentas de demo, shader e áudio

19. Farbrausch Public Repo

Roda em: Windows / Visual Studio antigo ou adaptação Linguagem: C++ Nível: Avançado

Descrição: Um baú bruto de ferramentas Farbrausch. Não espere tutorial fofinho. É arqueologia pesada e valiosa.

Aprenda: ferramentas históricas, kkrunchy, Werkkzeug, V2 synth, pipeline de demos

Abrir código/repositório
`https://github.com/farbrausch/fr_public`

20. Bonzomatic

Roda em: Windows, Linux, macOS; compila com CMake Linguagem: C++ / GLSL / HLSL Nível: Intermediário

Descrição: Um dos melhores jeitos de praticar shader como demoscener: escrever, recompilar, ver na hora.

Aprenda: live coding de shader, fragment shader em tempo real, feedback visual rápido

Abrir código/repositório
`https://github.com/Gargaj/Bonzomatic`

21. Rocket

Roda em: Windows / Linux; editor + lib C Linguagem: C / C++ / Qt Nível: Intermediário

Descrição: Ferramenta clássica para sincronizar variáveis do visual com música. Essencial para entender demo como peça audiovisual.

Aprenda: sincronização de música e visual, timeline, keyframes

Abrir código/repositório
`https://github.com/rocket/rocket`

22. Shader Minifier

Roda em: Windows; terminal/CMD; pode integrar ao build Linguagem: F# / shader tooling Nível: Avançado

Descrição: Ferramenta para reduzir shader sem perder comportamento. Serve para estudar como demosceners espremem código visual.

Aprenda: minificação GLSL/HLSL, preparação para 4k/64k, build pipeline

Abrir código/repositório
`https://github.com/laurentlb/Shader_Minifier`

23. SDF - Signed Distance Field Resources

Roda em: Leitura + exemplos Shadertoy Linguagem: GLSL / referências Nível: Intermediário

Descrição: Coleção boa para aprender a técnica que alimenta muita demo moderna: modelar formas por função matemática.

Aprenda: raymarching, campos de distância, geometria procedural

Abrir código/repositório
`https://github.com/CedricGuillemet/SDF`

24. Dwitter

Roda em: Browser / Python Django para servidor Linguagem: JavaScript / Python Nível: Intermediário

Descrição: A ideia é criar demos visuais minúsculas. Excelente treino para síntese visual e economia de código.

Aprenda: micro-demos JS, code golf visual, animações curtíssimas

Abrir código/repositório
`https://github.com/lionleaf/dwitter`

25. frame.js

Roda em: Browser / Node opcional Linguagem: JavaScript Nível: Intermediário

Descrição: Editor/estrutura JavaScript para organizar animações. Serve como laboratório para demos pequenas no navegador.

Aprenda: sequenciamento, timeline visual, animação em browser

Abrir código/repositório
`https://github.com/mrdoob/frame.js`

26. Axiom

Roda em: Windows / macOS; build C++ Linguagem: C++ Nível: Avançado

Descrição: Áudio também é código. Este repo ajuda a entender síntese sonora procedural, uma peça central em intros pequenas.

Aprenda: sintetizador node-based, áudio procedural, som para demos

Abrir código/repositório
`https://github.com/monadgroup/axiom`

27. OpenMPT / libopenmpt

Roda em: Windows; openmpt123 em terminal Linguagem: C++ Nível: Intermediário

Descrição: Tracker e biblioteca para música de módulos. Se você quer som retrô/demoscene, precisa entender esse ecossistema.

Aprenda: trackers, MOD/XM/S3M/IT, playback de música em demos e jogos

Abrir código/repositório
`https://github.com/OpenMPT/openmpt`

3 - PowerShell, CMD e animação de terminal

28. PowerShell Matrix Animation

Roda em: PowerShell 5.1+ / Windows Terminal / CMD chamando pwsh Linguagem: PowerShell Nível: Iniciante

Descrição: Comece aqui se quer algo que rode direto em PowerShell. Edite densidade, velocidade, renderizador e caracteres.

Aprenda: cursor, cores, parâmetros, loop de renderização no terminal

Abrir código/repositório
`https://github.com/avdaredevil/matrix`

29. fleschutz PowerShell Scripts

Roda em: PowerShell no Windows, Linux ou macOS Linguagem: PowerShell Nível: Iniciante

Descrição: Coleção enorme de scripts. Para estética terminal, procure conversão de imagem para ASCII, sons e scripts interativos.

Aprenda: scripts .ps1 autônomos, ASCII, som, utilidades, automação

Abrir código/repositório
`https://github.com/fleschutz/PowerShell`

30. clippy

Roda em: CMD no Windows; Python padrão Linguagem: Python Nível: Iniciante

Descrição: Muito bom para aprender o princípio básico: cada arquivo texto é um frame. Edite arte sem brigar com gráfico complexo.

Aprenda: frames ASCII em arquivos texto, animação simples, terminal screensaver

Abrir código/repositório
`https://github.com/con-dog/clippy`

31. ascii-animator

Roda em: PowerShell/CMD com Python Linguagem: Python Nível: Iniciante a intermediário

Descrição: Ferramenta e biblioteca para transformar GIFs em animações ASCII. Serve para aprender pipeline imagem -> texto -> tempo.

Aprenda: converter GIF para ASCII, animações, Game of Life, barras/equalizador

Abrir código/repositório
https://github.com/soda480/ascii-animator

32. Terminal Media Player - tplay

Roda em: Windows/Linux/macOS terminal Linguagem: Go Nível: Intermediário

Descrição: Media player ASCII. Estude como mapear luminância de pixels para caracteres e desenhar em alta velocidade no terminal.

Aprenda: imagem, vídeo e webcam em ASCII, renderização rápida no terminal

Abrir código/repositório
https://github.com/maxcurzi/tplay

33. CMatrix

Roda em: WSL/Linux; ports no Windows Linguagem: C Nível: Iniciante

Descrição: Clássico efeito Matrix no terminal. O código é simples o suficiente para hackear cores, velocidade e símbolos.

Aprenda: efeito Matrix, ncurses, timing e redesenho de tela

Abrir código/repositório
https://github.com/abishekvashok/cmatrix

34. TMatrix

Roda em: Terminal moderno; Windows via WSL ou build compatível Linguagem: Rust Nível: Intermediário

Descrição: Versão mais moderna e focada em precisão do digital rain. Boa para estudar performance e customização.

Aprenda: Matrix rain performático, terminal rendering, configuração

Abrir código/repositório
https://github.com/M4444/TMatrix

35. cxxmatrix

Roda em: Terminal ANSI; WSL recomendado no Windows Linguagem: C++ Nível: Intermediário

Descrição: Vai além do efeito Matrix: inclui cenas, banners, Game of Life e Mandelbrot. Um laboratório visual de terminal.

Aprenda: Matrix rain, Conway, Mandelbrot, cenas alternadas

Abrir código/repositório
https://github.com/akinomyoga/cxxmatrix

36. digi-rain

Roda em: Node/TypeScript em terminal Linguagem: TypeScript Nível: Intermediário

Descrição: Bom para quem quer editar animação terminal usando ecossistema Node em vez de C/C++.

Aprenda: efeito digital rain moderno, CLI JS, desenho incremental

Abrir código/repositório
https://github.com/gfargo/digi-rain

37. fakesteak

Roda em: C em terminal Unix/WSL Linguagem: C Nível: Iniciante

Descrição: Implementação enxuta de Matrix rain. É melhor para leitura de código que para frescura visual.

Aprenda: screensaver Matrix simples, terminal ANSI, distribuição minimalista

Abrir código/repositório
https://github.com/domsson/fakesteak

38. rusty-rain

Roda em: Rust em terminal cross-platform Linguagem: Rust Nível: Intermediário

Descrição: Use para aprender Rust aplicado a animação simples, em vez de exemplos abstratos e chatos.

Aprenda: CLI Rust, animação, render loop, caracteres/emoji

Abrir código/repositório
https://github.com/cowboy8625/rusty-rain

39. ASCII-renderer

Roda em: Linux/WSL; ncurses Linguagem: C Nível: Intermediário a avançado

Descrição: Renderiza objetos 3D no terminal usando caracteres. Excelente ponte entre álgebra linear e visual retrô.

Aprenda: renderizador 3D ASCII, câmera, iluminação, OBJ mesh

Abrir código/repositório
https://github.com/ShakedAp/ASCII-renderer

40. ASCII Donut Animation

Roda em: C++ em CMD/PowerShell após compilar Linguagem: C++ Nível: Iniciante a intermediário

Descrição: O famoso donut giratório. Se você quer entender matemática visual com payoff rápido, este é obrigatório.

Aprenda: donut 3D, seno/cosseno, z-buffer simplificado, ANSI colors

Abrir código/repositório
https://github.com/sherwinvishesh/ASCII-Donut-Animation

41. AsciiCube

Roda em: Browser; editável localmente Linguagem: JavaScript Nível: Iniciante

Descrição: Um cubo ASCII no navegador. Bom para estudar geometria sem toolchain pesada.

Aprenda: cubo 3D, Bresenham, rotação, projeção

Abrir código/repositório
https://github.com/jehna/AsciiCube

42. ascii-3d-cube

Roda em: PowerShell/CMD com Python Linguagem: Python Nível: Iniciante

Descrição: Versão Python de cubo 3D ASCII. Mais amigável para mexer do que C++ se você quer aprender pela tentativa.

Aprenda: cubo 3D, gradiente, profundidade, loop interativo

Abrir código/repositório
https://github.com/msadeqsirjani/ascii-3d-cube

43. ASCII-generator

Roda em: PowerShell/CMD com Python Linguagem: Python / OpenCV Nível: Intermediário

Descrição: Ferramenta para transformar mídia em ASCII. Dá para adaptar para animações, filtros e demos terminal.

Aprenda: converter imagem e vídeo para ASCII, OpenCV, mapeamento de brilho

Abrir código/repositório
https://github.com/vietnh1009/ASCII-generator

4 - Jogos e engines de terminal

44. FTXUI

Roda em: Windows/Linux/macOS terminal Linguagem: C++ Nível: Intermediário

Descrição: Biblioteca C++ para interfaces terminal. Não é demo pronta, mas é excelente base para jogos/visuais em console.

Aprenda: TUI moderna, layout, eventos, componentes visuais

Abrir código/repositório
https://github.com/ArthurSonzogni/FTXUI

45. ruscii

Roda em: Terminal; Rust Linguagem: Rust Nível: Intermediário

Descrição: Engine para criar jogos no terminal. Ótima para sair de animação passiva e entrar em jogo interativo.

Aprenda: engine gráfica terminal, input, sprites, jogos ASCII

Abrir código/repositório
https://github.com/lemunozm/ruscii

46. dotnet-console-games

Roda em: CMD/PowerShell com .NET Linguagem: C# Nível: Iniciante a intermediário

Descrição: Coleção educacional da própria comunidade .NET. Muito boa para aprender loop de jogo sem engine gráfica.

Aprenda: Snake, Pac-Man, 2048, Sudoku, Wordle, arquitetura de jogos console

Abrir código/repositório
https://github.com/dotnet/dotnet-console-games

47. CSharpConsoleGames

Roda em: CMD/PowerShell com .NET Linguagem: C# Nível: Iniciante

Descrição: Jogos clássicos em C# para console. Simples, editáveis, bons para entender estado, colisão e input.

Aprenda: Tetris, Snake, Tron, Pong, carros, desenho em console

Abrir código/repositório
https://github.com/NikolayIT/CSharpConsoleGames

48. cli-games

Roda em: Node.js em terminal Linguagem: JavaScript Nível: Iniciante

Descrição: Coleção de jogos terminal em JavaScript. Boa para prototipar rápido no CMD/PowerShell com Node.

Aprenda: jogos simples em CLI, menus, input, lógica de jogo

Abrir código/repositório
https://github.com/Seniru/cli-games

49. tetrigo

Roda em: Terminal com Go Linguagem: Go Nível: Intermediário

Descrição: Tetris em terminal escrito em Go. Bom se você quer aprender uma arquitetura CLI mais moderna.

Aprenda: Tetris, Bubble Tea/TUI, input e render loop

Abrir código/repositório
https://github.com/Broderick-Westrope/tetrigo

50. samtay/tetris

Roda em: Terminal; Windows por fonte/WSL Linguagem: Haskell Nível: Intermediário a avançado

Descrição: Tetris em Haskell. Não é o caminho fácil, mas ensina outra forma de pensar estado e renderização.

Aprenda: Tetris funcional, TUI Brick/Vty, arquitetura funcional

Abrir código/repositório
https://github.com/samtay/tetris

51. pokete

Roda em: Terminal; Python Linguagem: Python Nível: Intermediário

Descrição: Jogo grande em terminal. Bom para estudar como um projeto ASCII deixa de ser brinquedo e vira jogo de verdade.

Aprenda: RPG estilo Pokemon no terminal, mapas, sprites ASCII, estados

Abrir código/repositório
`https://github.com/lxgr-linux/pokete`

52. csol

Roda em: Terminal / DOS / ncurses ou pdcurses Linguagem: C Nível: Intermediário

Descrição: Coleção de paciências para terminal. Bom exemplo de TUI limpa e jogo tradicional sem gráficos modernos.

Aprenda: solitaire no terminal, interface em texto, input portável

Abrir código/repositório
`https://github.com/nielssp/csol`

5 - DOSBox, x86, VGA e retrocoding

53. masm-tasm

Roda em: VS Code + DOSBox/JSDOS/msdos-player Linguagem: Assembly 8086 Nível: Iniciante a intermediário

Descrição: Um dos pontos de entrada mais práticos para Assembly DOS hoje. Configura o ciclo editar-compilar-rodar.

Aprenda: rodar e debugar MASM/TASM, fluxo 8086 moderno

Abrir código/repositório
`https://github.com/dosasm/masm-tasm`

54. dos-dev-template

Roda em: VS Code + DJGPP + DOSBox-X Linguagem: C / C++ Nível: Intermediário

Descrição: Template excelente para criar app/jogo DOS com ferramenta moderna e debug decente. Muito mais racional que sofrer no escuro.

Aprenda: template DOS moderno, GDB remoto, modo 13h, assets

Abrir código/repositório
`https://github.com/badlogic/dos-dev-template`

55. dos3d

Roda em: DOSBox Linguagem: C Nível: Intermediário a avançado

Descrição: Renderização 3D em DOS. Se seu objetivo é entender gráficos de baixo nível, este é bem alinhado.

Aprenda: renderizador software para DOS, modo 13h, base estilo Doom/Quake

Abrir código/repositório
`https://github.com/kondrak/dos3d`

56. 54-byte Snake

Roda em: DOSBox / DOS Linguagem: x86 Assembly Nível: Avançado

Descrição: Snake em 54 bytes. É absurdo, mas ótimo para entender o espírito cruel e elegante do sizecoding.

Aprenda: sizecoding extremo, .COM tiny, jogo em pouquíssimos bytes

Abrir código/repositório
`https://github.com/donno2048/snake`

57. Tetris Assembly 8086

Roda em: EMU8086 para compilar; DOSBox para rodar Linguagem: Assembly 8086 Nível: Intermediário

Descrição: Tetris em Assembly com instruções de execução em DOSBox. Melhor que exemplos soltos porque tem jogo completo.

Aprenda: jogo completo em 8086, tela, input, lógica de peças

Abrir código/repositório
https://github.com/mkh2097/Tetris-Assembly-8086

58. Joe Wing DOS Programs

Roda em: NASM + DOSBox Linguagem: 386 Assembly Nível: Intermediário

Descrição: Coleção de jogos em Assembly DOS com código disponível. Excelente para retrocoding sem projeto gigantesco.

Aprenda: jogos DOS pequenos, Tetris, Lights Out, puzzle, montagem com NASM

Abrir código/repositório
https://joewing.net/projects/dos/

59. Wolfenstein 3D source

Roda em: Borland C++/ambiente antigo; estudo de código Linguagem: C / x86 / DOS Nível: Avançado

Descrição: Código original de Wolfenstein 3D. É histórico e educativo, mas não é plug-and-play moderno.

Aprenda: raycasting, VGA, loop de jogo, engine FPS antiga

Abrir código/repositório
https://github.com/id-Software/wolf3d

60. DOOM Open Source Release

Roda em: Linux build original; ports modernos para Windows Linguagem: C Nível: Avançado

Descrição: Código do Doom é leitura obrigatória para engine. Atenção: assets comerciais não vêm junto e a release oficial não é o DOS original.

Aprenda: BSP, renderização de Doom, arquitetura de engine, assets separados

Abrir código/repositório
https://github.com/id-Software/DOOM

61. DOS Games Archive - Source Code Games

Roda em: DOSBox Linguagem: C / Pascal / Assembly / variados Nível: Variado

Descrição: Página com jogos DOS que têm código-fonte. Use como garimpo, sempre conferindo licença e idade do projeto.

Aprenda: jogos DOS com fonte, comparação de estilos antigos

Abrir código/repositório
https://www.dosgamesarchive.com/license/source-code

62. DOS Haven Sources

Roda em: DOSBox / FreeDOS Linguagem: C / Pascal / ASM Nível: Iniciante a avançado

Descrição: Coleção de links para fontes e tutoriais DOS. Bom para montar seu currículo de retrocoding.

Aprenda: VGA, mode 13h, tutoriais e exemplos de programação DOS

Abrir código/repositório
https://www.doshaven.eu/sources/

6 - Emuladores e pontes úteis

63. js-dos

Roda em: Browser / JavaScript Linguagem: TypeScript / C++ via DOSBox Nível: Intermediário

Descrição: Útil quando você quiser transformar uma demo DOS ou jogo em algo que roda no browser.

Aprenda: rodar programas DOS no navegador, empacotar experiências retro

Abrir código/repositório
`https://github.com/caiiiycuk/js-dos`

64. DOSBox Staging

Roda em: Windows/Linux/macOS Linguagem: C++ Nível: Intermediário

Descrição: Emulador moderno para rodar e testar demos/jogos DOS. É ferramenta base para a parte DOSBox da sua trilha.

Aprenda: emulação DOS, áudio, vídeo, configuração, compatibilidade

Abrir código/repositório
`https://github.com/dosbox-staging/dosbox-staging`

65. 1oom

Roda em: Terminal/SDL; requer dados originais do jogo Linguagem: C Nível: Avançado

Descrição: Engine recreation de Master of Orion clássico. Boa para estudar portabilidade e reimplementação de sistemas antigos.

Aprenda: recriação de engine, estratégia por turnos, compatibilidade retro

Abrir código/repositório
`https://github.com/1oom-fork/1oom`

Fontes consultadas e pontos de validação

A lista foi montada a partir de páginas de repositórios, tópicos do GitHub, listas demoscene e arquivos de código-fonte DOS. Ainda assim, antes de investir horas em um projeto, confira licença, instruções de build, data de manutenção e issues abertas.

- Awesome Demoscene: https://github.com/psykon/awesome-demoscene
- Teach Yourself Demoscene in 14 Days: https://github.com/psenough/teach_yourself_demoscene_in_14_days
- Demoscene Starter Kits: https://github.com/anttihirvonen/demoscene-starter-kits
- Second Reality source: https://github.com/mtuomi/SecondReality
- GitHub topic: ascii-animation: https://github.com/topics/ascii-animation
- GitHub topic: terminal-animation: https://github.com/topics/terminal-animation
- GitHub topic: dosbox: https://github.com/topics/dosbox
- DOS Games Archive - source code games: https://www.dosgamesarchive.com/license/source-code
- DOS Haven - sources: https://www.doshaven.eu/sources/
- Bonzomatic: https://github.com/Gargaj/Bonzomatic
- Farbrausch fr_public: https://github.com/farbrausch/fr_public

Próximo passo prático

Escolha três projetos: um de PowerShell/CMD, um de ASCII 3D, e um de DOSBox. Faça uma modificação visível em cada um: trocar paleta, acelerar o loop, trocar caracteres, alterar input ou adicionar uma tela de créditos. Isso vale mais do que abrir cinquenta repositórios e não compilar nenhum.