Sites, links, descrição e dicas para efeitos JS em site com arte ANSI

PDF compilado a partir de pesquisa web atualizada e da leitura dos seus dois arquivos de referência (doom_fire.js e fire.js). O foco aqui é separar o que vale testar primeiro do que é só enfeite caro.

Resumo brutal

Para o seu caso, as referências mais alinhadas são: CodePen para protótipo rápido, OpenProcessing para sketch procedural, Codrops para tutorial sério, Three.js AsciiEffect e aalib.js quando você quer ASCII de verdade, e Shadertoy quando você quer shader pesado. Já os seus dois arquivos apontam para duas famílias claras: DOOM fire e fluid/fire simulation.

1. O que seus arquivos indicam

- Leitura dos seus arquivos enviados: doom_fire.js referencia explicitamente o artigo de Fabien Sanglard; fire.js referencia explicitamente o artigo de Andrew K. Chan.
- Fontes públicas priorizadas: páginas oficiais, documentação oficial e artigos/repositórios diretamente ligados aos efeitos.
- Data da coleta: 18 de março de 2026.

Leitura prática: o doom_fire.js segue a linha de overlay transparente sobre <pre>, com mapa sólido, paleta de fogo e propagação simples; o fire.js vai para uma simulação mais pesada, com campos de velocidade, temperatura, combustível e ruído. Ou seja: você está comparando um efeito retro-barato contra um efeito bonito-caro. É bom ter isso claro antes de sair coletando snippets aleatórios.

2. Lista principal de sites

Site	Melhor para	Link
CodePen	Demos rápidas / copiar e adaptar	https://codepen.io/
OpenProcessing	Creative coding / sketches p5.js	https://openprocessing.org/
GitHub Topics — ascii-art	Repositórios e bibliotecas reaproveitáveis	https://github.com/topics/ascii-art
Codrops	Tutoriais modernos / efeito explicado	https://tympanus.net/codrops/
Three.js AsciiEffect	ASCII gerado a partir de render 3D/WebGL	https://threejs.org/docs/pages/AsciiEffect.html
Shadertoy	Shaders prontos / estudo de GLSL	https://www.shadertoy.com/
GLSL.app	Sandbox leve para shader	https://glsl.app/
tsParticles	Partículas prontas para web	https://particles.js.org/
GSAP	Animação de interface / texto	https://gsap.com/docs/v3/GSAP/
aalib.js	Imagem/vídeo para ASCII	https://mir3z.github.io/aalib.js/

3. Detalhamento com uso prático

1. CodePen

Link: https://codepen.io/

O que é: Ambiente social de desenvolvimento focado em front-end, com editor no navegador para HTML, CSS e JavaScript.

Melhor uso no seu site: Melhor lugar para achar efeitos prontos, mexer em variáveis e testar rapidamente sobre um mock do seu ANSI.

Dica objetiva: Pesquise por: doom fire, ascii art, crt, scanlines, glitch, terminal, shader. Filtre pela aba Pens.

2. OpenProcessing

Link: https://openprocessing.org/

O que é: Comunidade de creative coding com editor intuitivo, sketches em p5.js e muitas referências em partículas, ruído, generative art e protótipos interativos.

Melhor uso no seu site: Bom para efeitos proceduralmente orgânicos, partículas, fumaça, glitches suaves e experimentos com tipografia/arte ASCII.

Dica objetiva: Procure por: particles, fire, terminal, noise, ascii. Aproveite o fluxo de fork para adaptar sketches.

3. GitHub Topics — ascii-art

Link: https://github.com/topics/ascii-art

O que é: Página de tópicos do GitHub para navegar por repositórios marcados com ascii-art e correlatos.

Melhor uso no seu site: Útil para achar bibliotecas, engines e projetos pequenos que você pode incorporar ou estudar antes de reescrever tudo do zero.

Dica objetiva: Além de ascii-art, vale buscar por terminal, ansi, crt, shader, dithering, postprocessing e image-to-ascii.

4. Codrops

Link: https://tympanus.net/codrops/

O que é: Portal de design e desenvolvimento web com tutoriais profundos, Creative Hub e demos de efeitos recentes.

Melhor uso no seu site: Excelente quando você quer entender como um efeito foi montado em vez de só colar snippet.

Dica objetiva: Entre também em Tutorials e Creative Hub; pesquise por Three.js, WebGL, typography, distortion, hover, ASCII e dithering.

5. Three.js AsciiEffect

Link: https://threejs.org/docs/pages/AsciiEffect.html

O que é: Addon oficial do Three.js que cria um efeito ASCII e substitui o domElement padrão do renderizador.

Melhor uso no seu site: Serve quando você quer transformar um render 3D, background WebGL ou cena procedural em ASCII sem inventar pipeline do zero.

Dica objetiva: Útil para backgrounds vivos por trás da sua arte ANSI, desde que você preserve contraste e legibilidade do texto.

6. Shadertoy

Link: https://www.shadertoy.com/

O que é: Plataforma para construir, compartilhar e explorar shaders; ótima para CRT, glow, heat haze, noise, distorção e pós-processamento.

Melhor uso no seu site: Melhor fonte para efeitos visuais pesados e shader-first: scanlines, bloom, dithering, fogo, plasma e glitches.

Dica objetiva: Use como laboratório e referência. Não copie cegamente: porte só a parte útil para o seu layout.

7. GLSL.app

Link: https://glsl.app/

O que é: Editor/sandbox online para WebGL/GLSL. A captura disponível retornou basicamente o título da ferramenta, então trate como ferramenta a validar com teste prático.

Melhor uso no seu site: Boa opção para prototipar shader pequeno sem acoplar tudo ao projeto principal logo de cara.

Dica objetiva: Use para provas de conceito curtas. Se ficar complexo, volte para Shadertoy ou integre num pipeline próprio.

8. tsParticles

Link: https://particles.js.org/

O que é: Biblioteca leve, browser-ready e compatível com vários frameworks para criar partículas, confete e fireworks.

Melhor uso no seu site: Serve como camada sutil de fundo; por exemplo, poeira, cinza, brasas ou partículas discretas atrás da ANSI art.

Dica objetiva: Use com moderação. Para ANSI, partículas demais viram ruído visual.

9. GSAP

Link: https://gsap.com/docs/v3/GSAP/

O que é: Ferramenta de animação JavaScript de nível profissional, com docs, demos e ecossistema forte.

Melhor uso no seu site: Perfeito para revelar linhas ANSI, mover blocos de texto, piscar cursor, transições de terminal e microinterações.

Dica objetiva: Primeiro resolva motion com GSAP/CSS. Só depois adicione shader ou canvas.

10. aalib.js

Link: https://mir3z.github.io/aalib.js/

O que é: Biblioteca JavaScript para converter imagens e filmes em ASCII no navegador.

Melhor uso no seu site: Boa quando você quer usar imagem ou vídeo como matéria-prima e transformá-los em ASCII antes de aplicar outros efeitos.

Dica objetiva: Mais útil para conversão e pipeline de ASCII do que para pós-processamento visual puro.

4. Referências diretamente alinhadas aos seus arquivos

Fabien Sanglard — How DOOM fire was done

Link: https://fabiensanglard.net/doom_fire_psx/

O que entrega: Explica a lógica do fogo do DOOM com framebuffer, paleta e propagação vertical com aleatoriedade e vento.

Por que importa no seu caso: Casa diretamente com o seu arquivo doom_fire.js.

filipedeschamps/doom-fire-algorithm

Link: https://github.com/filipedeschamps/doom-fire-algorithm

O que entrega: Playground em JavaScript com várias implementações e experimentos do efeito DOOM fire.

Por que importa no seu caso: Bom para pegar variantes prontas e comparar render em table, canvas, WASM e outras abordagens.

Andrew K. Chan — Simulating Fluids, Fire, and Smoke in Real-Time

Link: https://andrewkchan.dev/posts/fire.html

O que entrega: Texto técnico sobre simulação de fluidos/fogo/fumaça em tempo real com GPU/WebGL, combustão, buoyancy e renderização.

Por que importa no seu caso: Casa diretamente com o seu arquivo fire.js.

5. Termos de busca que valem mais do que “effects js”

- **ansi art overlay javascript**
- **doom fire algorithm js**
- **crt terminal effect webgl**
- **ascii distortion shader**
- **terminal glitch effect**
- **scanline shader webgl**
- **image to ascii javascript**
- **dithering shader three.js**
- **heat haze shader webgl**
- **p5.js fire particles ascii**

6. Ordem certa para testar

• Comece com 1 efeito principal e 1 efeito secundário. Mais do que isso normalmente destrói a legibilidade da ANSI art.

• Para site com cara terminal, GSAP + CSS resolve muito mais do que parece: reveal por linha, cursor, flicker, scroll parallax mínimo e entrada/saída de blocos.

• Se o foco for fogo retrô, prefira algoritmo estilo DOOM com overlay transparente sobre o . É barato e coerente com ANSI.

• Se o foco for “efeito bonito”, não vá direto para fluid simulation. Shader e fluido são mais bonitos, mas também mais caros, mais frágeis e mais fáceis de exagerar.

• Converta mídia para ASCII só quando isso fizer sentido narrativo. Se a arte já é ANSI boa, não estrague com filtro redundante.

• Teste contraste em fundo escuro e em densidades reais de texto. Efeito bonito em demo isolada costuma ficar ruim em layout real.

7. Conclusão honesta

Se a meta é deixar a ANSI art mais viva sem virar carnaval visual, a ordem racional é: GSAP/CSS para motion, depois CodePen/OpenProcessing para protótipos, depois DOOM fire se você quiser uma camada retrô coerente, e só então shader/fluid se realmente valer o custo. A maioria dos sites erra porque empilha efeitos demais e mata justamente o que deveria destacar: a arte.

8. Fontes usadas nesta coleta

- CodePen About: https://codepen.io/about
- OpenProcessing: https://openprocessing.org/
- GitHub Topics — ascii-art: https://github.com/topics/ascii-art
- Codrops About: https://tympanus.net/codrops/about/
- Three.js AsciiEffect docs: https://threejs.org/docs/pages/AsciiEffect.html
- Shadertoy: https://www.shadertoy.com/
- GLSL.app: https://glsl.app/
- tsParticles docs: https://particles.js.org/docs/index.html
- GSAP docs: https://gsap.com/docs/v3/GSAP/
- aalib.js: https://mir3z.github.io/aalib.js/
- Fabien Sanglard — How DOOM fire was done: https://fabiensanglard.net/doom_fire_psx/
- filipedeschamps/doom-fire-algorithm: https://github.com/filipedeschamps/doom-fire-algorithm
- Andrew K. Chan — Simulating Fluids, Fire, and Smoke in Real-Time: https://andrewkchan.dev/posts/fire.html