Relatorio em PDF com referencias

Onde conseguir mais JS como os seus scripts de fluido, fogo e ANSI

Pesquisa orientada para efeitos interativos de pagina em HTML/JS, com foco em bibliotecas, galerias de exemplos, editores e recursos para achar animacoes do tipo shader, particulas, canvas, WebGL/WebGPU, ASCII e terminal.

Resumo brutal: para achar mais coisas parecidas com os seus arquivos, pare de procurar “animacao JS” de forma generica. O seu terreno real e creative coding, shaders, particulas e textmode. O que mais vai te servir nao e biblioteca de tween DOM; e ecossistema de WebGL/WebGPU, p5.js, PixiJS, terminal ANSI e galerias de demos tecnicas.

Recomendacao curta

Se voce quer algo no mesmo espirito dos seus scripts: use p5.js + p5.asciify para prototipos ASCII, PixiJS para 2D acelerado, e Three.js / OGL / regl quando quiser subir o nivel para shaders e simulacoes mais pesadas. Para terminal/ANSI puro, use xterm.js, ansi_up e AnsiLove.js. Para achar codigo pronto e estudar tecnicas, o trio mais util e Codrops, CodePen e OpenProcessing.

1. Mapa rapido: o que usar em cada caso

Objetivo	Biblioteca / recurso	Nivel	Por que serve
Quero efeito pronto rapido	Vanta.js [R12], tsParticles [R11]	Baixo	Entrega backgrounds e particulas configuraveis sem voce escrever simulacao do zero.
Quero landing page forte com scroll e composicao	GSAP [R13], Anime.js [R14], Codrops [R15]	Baixo a medio	Controla timing, entradas, saidas, scroll e costura varios efeitos sem reinventar timeline.
Quero canvas 2D rapido	PixiJS [R6], @pixi/particle-emitter [R7]	Medio	Otimo para particulas, mascaras, filtros e composicao 2D com aceleracao GPU.
Quero shader / WebGL / fluid-like visuals	Three.js [R1], OGL [R2], regl [R3]	Medio a alto	Da controle real sobre pipeline grafico, post-processamento e simulacoes visuais mais serias.
Quero prototipo artistico e exploratorio	p5.js [R4], OpenProcessing [R17], p5 sketches [R5]	Baixo a medio	Perfeito para testar ideia, mexer em ruido, fluidos simplificados, input e estetica experimental.

Objetivo	Biblioteca / recurso	Nivel	Por que serve
Quero manter a cara ASCII / terminal	p5.asciify [R8], xterm.js [R9], ansi_up [R10], AnsiLove.js [R18]	Medio	Consegue renderizar, converter ou simular texto/ANSI em tempo real ou a partir de arquivos .ans.

2. As familias de bibliotecas que realmente importam

2.1 Three.js, OGL e regl: para sair do “efeito legal” e entrar em grafica real

Three.js tem ecossistema enorme, galeria oficial de exemplos e documentacao de post-processing; e o caminho mais seguro para efeitos de tela cheia, composicao 3D, materiais, feedback visual e transicoes mais serias [R1]. OGL e bem menor, com menos abstrações e foco em quem quer escrever shader e entender o que esta acontecendo [R2]. regl fica ainda mais perto do WebGL cru, com uma API funcional pensada para comandos e recursos GPU, bom para quem quer controle sem aceitar toda a estrutura de um engine [R3].

2.2 PixiJS: quando o problema e 2D acelerado e nao 3D

PixiJS hoje se posiciona como engine 2D rapida para WebGL e WebGPU, com filtros, blend modes, textos, mascaras e objetos de cena [R6]. Para paginas com particulas, faixas de energia, glow, filtros de distorcao, chuva de caracteres ou overlays estilizados, Pixi costuma ser mais leve e objetivo do que abrir um projeto Three.js inteiro. O emissor oficial de particulas adiciona um sistema dedicado para design de emissores [R7].

2.3 p5.js: o melhor lugar para prototipar ideia visual sem burocracia

p5.js continua sendo um dos melhores ambientes para sketch visual, ruido, interacao, testes de grade e experimentacao rapida [R4]. O valor real nao e “efeito pronto”; e velocidade de exploracao. Para quem esta mexendo com fluidos simplificados, campos vetoriais, ruido, trails, tipografia e resposta a mouse/teclado, p5.js costuma ser o jeito mais rapido de chegar num prototipo legivel.

2.4 p5.asciify, xterm.js, ansi_up e AnsiLove.js: sua trilha textmode

Se voce quer animacao com cara de terminal ou ASCII, existem caminhos diferentes. p5.asciify converte canvas e texturas WEBGL em representacoes ASCII em tempo real e ainda tem pagina de recursos com fontes e assets testados [R8]. xterm.js e um frontend de terminal real no browser, mais apropriado quando a interface deve se comportar como terminal de verdade [R9]. ansi_up converte sequencias ANSI em HTML colorido sem dependencias [R10]. AnsiLove.js renderiza formatos de textmode/scene como .ANS no cliente [R18].

2.5 Vanta.js, tsParticles, GSAP e Anime.js: produtividade, nao profundidade grafica

Vanta.js entrega backgrounds 3D prontos, mas o proprio site avisa que alguns efeitos sao lentos em hardware mais fraco e que nao convem usar muitos numa mesma pagina [R12]. tsParticles e um kit forte para particulas, com presets como fire, fireworks e fountain [R11]. GSAP e Anime.js resolvem a camada de animacao e sincronizacao: excelentes para orquestrar elementos, textos, SVG, canvas, WebGL e scroll [R13][R14]. Eles nao substituem shader/simulacao, mas fazem a cola do projeto.

3. Onde achar demos, codigo e inspiracao util

Lugar	O que voce acha la	Como usar sem perder tempo
Three.js examples [R1]	Exemplos oficiais de loaders, post-processing, materiais, camera, particulas, WebGPU e cenas completas.	Nao copie a demo inteira. Extraia so a tecnica: shader, post-process, controle de camera, feedback visual.
Codrops [R15]	Tutoriais modernos de WebGL, scroll, galerias, transicoes e misturas com GSAP/OGL/Three.js.	Excelente para aprender composicao de pagina e nao apenas o efeito isolado.
CodePen [R16]	Microdemos de front-end. Otimo para achar experimentos pequenos e testar rapido no navegador.	Use tags e filtros; busque por “ascii”, “particles”, “shader”, “webgl”, “pixi”, “fluid”.
OpenProcessing [R17]	Comunidade de creative coding focada em compartilhar e remixar projetos.	Muito bom para estudar sketchs de p5.js, ruido, trails, tipografia generativa e interacao.
p5.js sketches e libraries [R5]	Galeria de sketches e indice de bibliotecas que estendem p5.	Bom para achar tecnicas remixaveis e extensoes menores em vez de frameworks gigantes.
glsl.app e GLSL Sandbox [R19][R20]	Playgrounds de shader para testar fragment shaders, ruido, distorcoes, feedback e composicao procedural.	Ideal para estudar so a camada visual antes de integrar numa pagina real.
p5.textmode.art resources [R8]	Fontes textmode/pixel e assets voltados ao ecossistema p5.asciify.	Vale para achar fonte certa e evitar um ASCII bonito com tipografia errada.

Atalho pratico de busca

Busque sempre em ingles e por tecnicas, nao por resultado final. Exemplos de consultas que funcionam melhor: “webgl fluid distortion three.js”, “pixi particle glow trail”, “p5 ascii shader”, “xterm ansi animation”, “ogl shader gallery”, “codepen canvas fluid”, “openprocessing noise field”.

4. Pilhas recomendadas para o seu caso

- Pilha A - ASCII experimental com prototipagem rapida: p5.js + p5.asciify + OpenProcessing. Melhor para iterar ideia, achar fonte, testar comportamento de grade e chegar a um visual forte em pouco tempo [R4][R8][R17].
- Pilha B - Site serio com visual forte e GPU: Three.js + GSAP + Codrops como fonte de estudos. Melhor quando a pagina precisa de impacto visual, scroll coreografado e shaders de verdade [R1][R13][R15].
- Pilha C - 2D estilizado e particulas: PixiJS + @pixi/particle-emitter + filtros. Melhor para chuva de particulas, brasas, energia, glow e overlays 2D [R6][R7].
- Pilha D - Terminal / ANSI: xterm.js quando quiser comportamento de terminal, ansi_up para converter ANSI em DOM, e AnsiLove.js quando o ativo central for um arquivo .ans/.xb/.pcb [R9][R10][R18].
- Pilha E - efeito pronto com pouca engenharia: Vanta.js ou tsParticles, com GSAP para transicoes de entrada/saida. Serve para landing page rapida, mas nao substitui uma base grafica boa [R11][R12][R13].

5. Coisas que voce deve filtrar antes de copiar qualquer demo

- Licenca: muita demo bonita e codigo de vitrine, nao kit de producao. Verifique licenca e dependencia escondida.

- GPU vs CPU: seus scripts atuais ja mostram que simulacao em CPU pode ficar cara. Se o efeito crescer, migrate a parte pesada para shader ou engine 2D/3D mais apropriada.
- Mobile e reduced motion: sempre tenha fallback, opcao de desativar e modo mais leve. O proprio Vanta.js avisa sobre limites em maquinas lentas e mobile [R12].
- Bundle size: nao enfie Three.js se um canvas 2D ou Pixi resolve. Nao use um terminal completo se so precisa colorir ANSI.
- Integracao real: alguns recursos sao otimos para estudo (GLSL Sandbox, glsl.app), mas nao sao a solucao de producao final. Sao laboratorios.
- Fonte e rasterizacao: em ASCII/textmode, a tipografia correta muda tudo. Isso nao e detalhe estetico; e parte do render [R8][R18].

6. Fluxos de trabalho que economizam tempo

1. Prototipe o comportamento em p5.js ou CodePen.
2. Se a ideia funcionar, reescreva o nucleo no stack certo: PixiJS para 2D, Three/OGL/regl para shader/WebGL.
3. Se o visual final for textmode, decida cedo se o render e ASCII “simulado” (p5.asciify, ansi_up) ou ANSI/classico (.ans) com renderer dedicado (AnsiLove.js).
4. Use GSAP ou Anime.js so para sincronizar interfaces, paines, entradas, saidas, textos e scroll; nao para fingir simulacao fisica [R13][R14].

7. Instalacoes iniciais uteis

Three.js + GSAP

```
npm i three gsap
```

PixiJS + emissor de particulas

```
npm i pixi.js @pixi/particle-emitter
```

p5.js + ASCII em tempo real

```
npm i p5 p5.asciify
```

Terminal / ANSI no browser

```
npm i @xterm/xterm ansi_up
```

Particulas prontas

```
npm i tsparticles
```

8. Conclusao direta

Se o objetivo for achar “mais JS como os seus”, a resposta correta nao e uma lista aleatoria de plugins. O caminho certo e separar por camada: efeito pronto, motor 2D, motor 3D/shader, terminal/ANSI e lugar de pesquisa. Para o seu perfil, a combinacao mais promissora e p5.js + p5.asciify para explorar, PixiJS para 2D forte e Three.js/OGL/regl para efeitos que precisem de GPU de verdade. Use CodePen, Codrops e OpenProcessing para descoberta. Use GSAP ou Anime.js so como coreografia, nao como motor visual.

Decisao recomendada

Se eu tivesse que cortar a lista e te deixar so com tres pontos de partida: (1) p5.js + p5.asciify para explorar rapido; (2) PixiJS se a pagina for mais 2D/particulas; (3) Three.js + GSAP se a homepage tiver que parecer produto serio e visualmente agressivo.

9. Referencias

R1 - Three.js examples; Three.js post-processing docs; manual de WebGPU post-processing - https://threejs.org/examples/ ; https://threejs.org/docs/pages/PostProcessing.html ; https://threejs.org/manual/en/webgpu-postprocessing.html

R2 - OGL official site and repo - https://oframe.github.io/ogl/ ; https://github.com/oframe/ogl

R3 - regl docs and project page - https://regl-project.github.io/regl/

R4 - p5.js reference - https://p5js.org/reference/

R5 - p5.js sketches and libraries - https://p5js.org/sketches/ ; https://p5js.org/libraries/

R6 - PixiJS guides for filters and particle containers - https://pixijs.com/8.x/guides/components/filters ; https://pixijs.com/8.x/guides/components/scene-objects/particle-container

R7 - @pixi/particle-emitter docs - https://particle-emitter.pixijs.io/docs/

R8 - p5.asciify repo and community resources - https://github.com/humanbydefinition/p5.asciify ; https://p5.textmode.art/docs/community/resources/

R9 - xterm.js docs - https://xtermjs.org/docs/

R10 - ansi_up repo and npm package - https://github.com/drudru/ansi_up ; https://www.npmjs.com/package/ansi_up

R11 - tsParticles docs - https://particles.js.org/docs/

R12 - Vanta.js site and repo - https://www.vantajs.com/ ; https://github.com/tengbao/vanta

R13 - GSAP docs - https://gsap.com/docs/v3/

R14 - Anime.js docs - https://animejs.com/documentation/

R15 - Codrops WebGL / Tutorials - https://tympanus.net/codrops/tag/webgl/ ; https://tympanus.net/codrops/category/tutorials/

R16 - CodePen features and examples - https://codepen.io/features ; https://codepen.io/examples ; https://codepen.io/

R17 - OpenProcessing community guidance - https://preview.openprocessing.org/home/comg

R18 - AnsiLove.js site and downloads - https://ansilove.github.io/ansilove.js/ ; https://www.ansilove.org/downloads.html

R19 - glsl.app - https://glsl.app/

R20 - GLSL Sandbox / Experiments with Google - https://glslsandbox.com/e ; https://experiments.withgoogle.com/glsl-sandbox