Construindo um site inteiro com .ANS, HTML e JavaScript

Relatório técnico

Construindo um site inteiro com .ANS, HTML e JavaScript

Panorama prático para montar um site com estética ANSI/textmode: formatos, bibliotecas JS, softwares de criação, fontes, acervos, fluxo de produção, limitações reais e uma stack recomendada.

Resumo brutalmente honesto

Fazer um site *inteiro* em ANSI puro é possível, mas é uma má ideia para navegação, SEO, acessibilidade e responsividade. O caminho profissional é usar HTML/CSS normal para estrutura e reservar ANSI para hero, banners, menus, telas especiais e experiências interativas.

Decisão principal

Se o foco é **.ANS clássico / CP437 / SAUCE**, use **AnsiLove.js**. Se o foco é **terminal web vivo** e sequências modernas, use **xterm.js**. Se o foco é apenas converter ANSI colorido para DOM, use **ansi_up**.

Preparado em 7 de março de 2026

Idioma: português

Com referências e links de origem

Visão geral

A confusão mais comum é tratar "ANSI" como uma coisa só. Na prática, existem pelo menos dois mundos diferentes: o **ANSI art clássico** ligado a BBS/MS-DOS, CP437 e convenções como SAUCE; e o **ANSI moderno de terminal**, baseado em sequências de escape e SGR, incluindo 256 cores e truecolor 24-bit ESC[38;2;R;G;Bm / ESC[48;2;R;G;Bm.[16][18]

Conteúdo

Conclusão rápida

Para um projeto sério: faça o site em HTML/CSS/JS normal e injete ANSI onde ele agrega identidade visual. Tentar transformar cada página em um pseudo-terminal é charmoso por cinco minutos e cansativo depois.

1) O que realmente significa "fazer um site com .ANS"

Você pode seguir três estratégias totalmente diferentes, e escolher a errada logo no início é o jeito mais fácil de produzir algo bonito e inútil.

Galeria textmode clássica .ANS / .ASC / .BINCP437SAUCE

Ideal para portfolio, artpacks, banners, NFOs e telas estáticas. O render mais natural no browser é via **AnsiLove.js**, que desenha o arquivo em *canvas* e suporta formatos da cena como ANS, PCB, BIN, ADF, IDF, TND e XB.[1]

Terminal web "de verdade" TTY feelinputaddons

Ideal para shell fake, boot screens, sessões interativas, experiências "hacker aesthetic", logs vivos e demos. O caminho certo é **xterm.js**, um frontend de terminal para browser, com docs de sequências, parser e addons.[2][9]

ANSI colorido convertido em HTML SGRDOMlogs

Ideal para blocos simples coloridos dentro de páginas comuns. **ansi_up** converte texto ANSI em span estilizado, mas só interpreta SGR e URLs escapadas; códigos de movimento de cursor são ignorados.[3]

Armadilha conceitual

Nem todo arquivo "ANSI" tem a mesma semântica. O próprio ecossistema histórico tem variantes e extensões; não existe uma especificação única, limpa e universalmente seguida.[19] Se você presumir compatibilidade total entre visualizadores, vai quebrar arte, alinhamento e cor.

2) ANSI clássico vs ANSI moderno

ANSI clássico / BBS É a tradição de ANSI art distribuída como .ANS, exibida em consoles textmode e associada a Code Page 437. O arquivo pode trazer metadados SAUCE, e muitos visualizadores assumem 80 colunas, fontes textmode específicas e tratamento de iCE colors.[1][17][20]

ANSI moderno / terminal É o uso contemporâneo de sequências de escape/controle para colorir, mover cursor, alternar atributos e desenhar interfaces no terminal. O núcleo padronizado vem do ECMA-48, e a família SGR inclui códigos para reset, negrito, sublinhado, 8/16 cores, 256 cores e truecolor.[16][18]

Caso	Melhor opção	Por quê
Mostrar arte .ans fiel	AnsiLove.js	Renderiza formatos da cena direto no cliente; suporta SAUCE, iCE color
Simular terminal vivo	xterm.js	Foi feito para terminal no navegador; possui docs de sequências, parser e
Colorir logs/texto em página comum	ansi_up	Converte ANSI em HTML com segurança e pode usar classes CSS em ve

Arquiteturas possíveis e stack recomendada

3) A stack que faz sentido

Recomendação prática

HTML/CSS/JS normais para estrutura + **AnsiLove.js** para assets .ans clássicos + **xterm.js** para áreas interativas + **ansi_up** para logs ou trechos coloridos simples. Isso maximiza estética e minimiza dor de manutenção.[1][2][3][9]

4) Quando cada arquitetura faz sentido

Arquitetura	Boa para	Implementação	Ve
Site 100% terminal	Peça artística, microsite, gimmick	Tudo renderizado em terminal web; navegação imita shell	Fu
Site híbrido	Portfolio, produto, blog, documentação	HTML responsivo + blocos ANSI onde interessar	**M**
Galeria de arte ANSI	Arquivo, acervo, showcases	Assets .ans renderizados em canvas/imagem	Ex

5) Starter stack sugerida

- **Framework**: qualquer stack front-end serve; para simplicidade, HTML estático ou Astro/Next/Vite já resolvem.
- **Renderer clássico**: **AnsiLove.js** para .ans / .pcb / .bin / afins.[1]
- **Terminal interativo**: **xterm.js** + **@xterm/addon-fit** para caber no container responsivo.[2][9][10]
- **Blocos SGR**: **ansi_up** quando quiser apenas transformar ANSI colorido em DOM.[3]
- **Fontes**: use fontes PC oldschool compatíveis com webfont via @font-face; o pack da INT 10h é a melhor base pública para isso.[11][15]
- **Acervos**: 16colo.rs e Artpacks.org para referência, estudo e assets históricos.[12][13]

6) O que não fazer

- Não dependa do terminal para navegação, formulários, SEO e leitura longa.
- Não assuma que qualquer ANSI viewer vai renderizar igual.
- Não use fonte aleatória e depois culpe o parser pelo desalinhamento.
- Não misture ANSI clássico e ANSI truecolor como se fosse tudo o mesmo pipeline.
- Não incorpore webfonts sem checar licença; @font-face não te dá permissão jurídica por mágica.[15][21]

7) Estrutura de projeto recomendada

```
/public
  /ans         # arquivos .ans/.asc/.bin
  /fonts       # WOFF/WOFF2 de fontes textmode licenciadas
  /img         # PNGs ou screenshots auxiliares
/src
  /components
    AnsiCanvas.js     # wrapper do AnsiLove.js
    TerminalHero.js   # wrapper do xterm.js
    AnsiBlock.js      # wrapper do ansi_up
  /styles
    textmode.css
index.html
```

Regra simples

Se o conteúdo precisa ser indexável, acessível, selecionável, traduzível e responsivo, mantenha-o em HTML real. Use ANSI como camada estética ou de experiência, não como prisão arquitetural.

Softwares para criar e editar ANSI

8) Editores e utilitários que valem seu tempo

Ferramenta	Tipo	Pontos fortes
AnsiDraw	Editor web	Sem instalação, sem login, importa .ANS local ou por URL, exporta para .ANS e PNG, canva
Moebius	Desktop	Windows/macOS/Linux. O diferencial é o *half-block brush*, com fluxo mais próximo de editor
PabloDraw	Desktop	Editor/viewer ANSI/ASCII com capacidades multi-user; clássico e bastante conhecido na cena
Durdraw	Terminal app	Roda em terminais UTF-8, suporta animação, 16/256 cores, mistura CP437/Unicode, saída H
IcyDraw	Desktop moderno	Se apresenta como sucessor espiritual do MysticDraw, com workflow moderno para arte textm
AnsiLove	Conversor	Projeto maduro para converter ANSi e formatos da cena para PNG; útil para gerar thumbs, p

9) Onde conseguir arte, packs e referência visual

16colo.rs Arquivo de ANSI/ASCII artpacks preservando lançamentos desde o início dos anos 1990 até o presente. Excelente para estudar lettering, composição e estética de grupo/cena.[12]

Artpacks.org Arquivo de packs da cena BBS/ANSI/ASCII de 1990 até o presente. Útil tanto para estudo quanto para pesquisa histórica e referência tipográfica/visual.[13]

INT 10h Oldschool PC Fonts Maior coleção pública de fontes textmode clássicas remasterizadas em TTF/FON/WOFF, com mais de 200 conjuntos de caracteres.[11]

RetroTxt Extensão de navegador para ver ANSI/ASCII/NFO como documento web, com múltiplas fontes, paletas, parsing de SAUCE e vários encodings.[14]

Atalho útil

Se você só quer estudar visual e não montar pipeline ainda, use Artpacks/16colo para referência, RetroTxt para inspeção rápida e AnsiDraw para rascunhar sem instalar nada.[4][12][13][14]

10) Fontes: a parte que quase todo mundo subestima

ANSI art depende pesadamente da fonte certa. Se você usa uma fonte "parecida" em vez da fonte adequada, blocos, box-drawing, proporção vertical e sensação de textmode se degradam. A INT 10h mantém um acervo extenso de fontes clássicas em formatos adequados para web, e a regra @font-face é o mecanismo padrão para incorporar essas fontes em páginas web.[11][15]

```
@font-face {
  font-family: 'Px437 IBM VGA 8x16';
  src: url('/fonts/Px437_IBM_VGA_8x16.woff2') format('woff2');
}

.textmode {
  font-family: 'Px437 IBM VGA 8x16', monospace;
}
```

Licenciamento

Não assuma que comprar uma fonte desktop ou achar um ZIP em fórum te autoriza a publicar a fonte na web. O próprio material do ecossistema webfont alerta para verificar a licença de incorporação web antes de usar @font-face em produção.[21]

Técnicas web indispensáveis

11) Preserve o espaçamento do texto

Em HTML, ANSI/textmode sem preservação de whitespace vira bagunça. A propriedade CSS white-space controla exatamente isso.[23] Para arte textual e blocos preformatados, use pre ou pre-wrap com critério.

```
.ansi-block {
  white-space: pre;
  overflow-x: auto;
}

.ansi-block--wrap {
  white-space: pre-wrap;
}
```

12) Preserve pixels quando escalar imagens

Se você exibir thumbs/renderizações PNG de ANSI art, escalar sem controle costuma borrar tudo. A propriedade image-rendering com valor pixelated ajuda a preservar bordas duras e o aspecto de pixel art ao aumentar a imagem.[24]

```
.ansi-preview img {
  image-rendering: pixelated;
}
```

13) Entenda o limite de cada biblioteca

- **AnsiLove.js**: excelente para visualização fiel de arte e formatos da cena; ainda oferece splitRender para arquivos muito grandes.[1]

- **ansi_up**: muito bom para ANSI colorido simples, mas ignora movimento de cursor e, portanto, não substitui um terminal real.[3]
- **xterm.js**: é o cavalo de batalha quando existe interatividade, parser, sequências suportadas e addons.[2][9][10]

14) Não trate CP437 como "detalhe cosmético"

O ANSI clássico foi historicamente ligado ao IBM code page 437 e a outras codepages correlatas; diversos renderizadores e visualizadores explicitam suporte a charsets específicos. Se você converter bytes para Unicode de maneira ingênua ou escolher a fonte errada, o desenho quebra.[1][20]

15) Acessibilidade e SEO: onde o romantismo morre

Problema Canvas, imagens e pseudo-terminais são péssimos como recipiente de conteúdo principal quando o objetivo é legibilidade, indexação e tecnologias assistivas.

Saída adulta Mantenha títulos, textos, navegação e conteúdo importante em HTML semântico. Use ANSI como decoração funcional, bloco visual, arte de abertura ou interface opcional.

16) Performance e responsividade

Xterm.js possui addons para ampliar funcionalidade e um addon de *fit* para ajustar o terminal ao container.[9][10] Isso resolve uma parte do problema responsivo, mas não elimina o fato de que terminais têm grid rígido: em telas pequenas, a densidade de colunas continua sendo um problema de UX.

Regra de ouro de layout

No desktop, ANSI pode ocupar hero, seções inteiras e páginas de showcase. No mobile, ele deve degradar bem: scroll horizontal controlado, versão simplificada, screenshot/pôster estático ou bloco colapsável.

Pipeline de produção e exemplos mínimos

17) Fluxo recomendado

1. Escolha o tipo de ANSI que você realmente vai usar: **clássico** ou **moderno**.
2. Desenhe/edite a arte em AnsiDraw, Moebius, PabloDraw, Durdraw ou IcyDraw, conforme seu perfil.[4][5][6][7][8]
3. Padronize fontes web e fallback visual.
4. Decida se cada asset será renderizado no cliente (canvas/terminal) ou pré-renderizado como PNG para performance/consistência.
5. Monte a estrutura do site em HTML semântico.
6. Use ANSI apenas nas zonas onde ele realmente melhora a identidade do projeto.
7. Teste em desktop, mobile e pelo menos dois navegadores.
8. Valide degradação: sem webfont, sem JS e em viewport estreita.

18) Exemplo mínimo com AnsiLove.js

```
<div id="art"></div>
<script src="/js/ansilove.min.js"></script>
<script>
  AnsiLove.render('/ans/home.ans', function (canvas, sauce) {
    canvas.style.imageRendering = 'pixelated';
    document.getElementById('art').appendChild(canvas);
    console.log(sauce);
  }, {
    font: '80x25',
    bits: '8',
    icecolors: 1,
    columns: 80
  });
</script>
```

Esse caminho é o melhor quando você quer mostrar arquivos da cena e preservar parâmetros típicos de renderização, inclusive SAUCE e iCE colors.[1]

19) Exemplo mínimo com xterm.js

```
import { Terminal } from '@xterm/xterm';
import { FitAddon } from '@xterm/addon-fit';

const term = new Terminal({ cursorBlink: true });
const fit = new FitAddon();
term.loadAddon(fit);
term.open(document.getElementById('terminal'));
fit.fit();
term.write('\x1b[38;2;0;255;255mBEM-VINDO\x1b[0m\r\n');
```

Xterm.js foi feito exatamente para isso: terminal funcional no browser, com API, sequências suportadas e addons para extensão.[2][9][10]

20) Exemplo mínimo com ansi_up

```
import AnsiUp from 'ansi_up';

const ansi = new AnsiUp();
ansi.use_classes = true;
const html = ansi.ansi_to_html('\u001b[31mERRO\u001b[0m sistema online');
document.getElementById('log').innerHTML = html;
```

Isso é ótimo para caixas de log e trechos pequenos. Não tente forçar interfaces complexas aqui, porque ansi_up não interpreta movimento de cursor.[3]

21) Estratégia de fallback

Cenário	Fallback recomendado	Motivo
Sem JavaScript	PNG gerado previamente	Evita tela vazia e mantém identidade visual.
Sem webfont	Monospace + aviso visual discreto	O layout fica pior, mas ainda legível.
Mobile estreito	Scroll horizontal controlado ou versão simplificada	Grid terminal e 80 colunas não cabem magicament
SEO / leitores de tela	HTML semântico duplicando o conteúdo essencial	Canvas e terminal decorativo não substituem conte

Decisão final que eu tomaria

Se o objetivo é ter um site usável e não apenas uma peça de cena, eu faria um **site híbrido**: conteúdo e navegação em HTML real, branding e experiências especiais em ANSI. Qualquer tentativa de empurrar tudo para .ans puro vira fetish técnico, não produto bom.

Referências

1. AnsiLove.js. *Textmode-Art For Your Browser*. Documentação oficial. Disponível em: https://ansilove.github.io/ansilove.js/ . Acesso em: 2026-03-07.
2. xterm.js. *Documentation 6.0*. Documentação oficial. Disponível em: https://xtermjs.org/docs/ . Acesso em: 2026-03-07.
3. drudru. *ansi_up*. Repositório GitHub. Disponível em: https://github.com/drudru/ansi_up . Acesso em: 2026-03-07.
4. AnsiDraw. *Free Online ANSI Art Editor*. Disponível em: https://ansidraw.com/ . Acesso em: 2026-03-07.
5. Blocktronics. *Moebius ANSI Art Editor*. Disponível em: https://blocktronics.github.io/moebius/ . Acesso em: 2026-03-07.
6. Curtis Wensley. *PabloDraw*. Repositório GitHub. Disponível em: https://github.com/cwensley/pablodraw . Acesso em: 2026-03-07.

7. Durdraw. *ANSI, ASCII and Unicode Art Animation Studio for Linux*. Disponível em: https://durdraw.org/ . Acesso em: 2026-03-07.
8. mkrueger. *IcyDraw / icy_tools*. Repositório GitHub. Disponível em: https://github.com/mkrueger/icy_tools . Acesso em: 2026-03-07.
9. xterm.js. *Using addons*; *@xterm/addon-fit*. Disponível em: https://xtermjs.org/docs/guides/using-addons/ e https://www.npmjs.com/package/@xterm/addon-fit . Acesso em: 2026-03-07.
10. npm. *@xterm/xterm*. Disponível em: https://www.npmjs.com/package/@xterm/xterm . Acesso em: 2026-03-07.
11. INT 10h. *The Ultimate Oldschool PC Font Pack*. Disponível em: https://int10h.org/oldschool-pc-fonts/ . Acesso em: 2026-03-07.
12. 16colo.rs. *ANSI/ASCII art archive*. Disponível em: https://16colo.rs/ . Acesso em: 2026-03-07.
13. Artpacks.org. *ANSI & ASCII Art Archive*. Disponível em: https://artpacks.org/ . Acesso em: 2026-03-07.
14. RetroTxt. *Install, screenshots and features*. Disponível em: https://docs.retrotxt.com/ . Acesso em: 2026-03-07.
15. MDN Web Docs. *@font-face*. Disponível em: https://developer.mozilla.org/en-US/docs/Web/CSS/Reference/At-rules/@font-face . Acesso em: 2026-03-07.
16. Wikipedia. *ANSI escape code*. Disponível em: https://en.wikipedia.org/wiki/ANSI_escape_code . Acesso em: 2026-03-07.
17. ACiD. *SAUCE - Standard Architecture for Universal Comment Extensions*. Disponível em: https://www.acid.org/info/sauce/ . Acesso em: 2026-03-07.
18. Ecma International. *ECMA-48: Control functions for coded character sets*. Disponível em: https://ecma-international.org/publications-and-standards/standards/ecma-48/ . Acesso em: 2026-03-07.
19. ArchiveTeam / File Formats Wiki. *ANSI Art*. Disponível em: https://fileformats.archiveteam.org/index.php?title=ANSI_A . Acesso em: 2026-03-07.
20. Wikipedia. *ANSI art*. Disponível em: https://en.wikipedia.org/wiki/ANSI_art . Acesso em: 2026-03-07.
21. web.dev. *Quick guide to web fonts via @font-face*. Disponível em: https://web.dev/articles/webfonts-quick . Acesso em: 2026-03-07.
22. AnsiLove. *Downloads*. Disponível em: https://www.ansilove.org/downloads.html . Acesso em: 2026-03-07.
23. MDN Web Docs. *white-space*. Disponível em: https://developer.mozilla.org/pt-BR/docs/Web/CSS/Reference/Properties/white-space . Acesso em: 2026-03-07.
24. MDN Web Docs. *image-rendering*. Disponível em: https://developer.mozilla.org/en-US/docs/Web/CSS/Reference/Properties/image-rendering . Acesso em: 2026-03-07.

Este relatório prioriza fontes oficiais e documentação primária. Onde isso não existe de forma completa ou pública, foram usados acervos e páginas históricas relevantes para o ecossistema ANSI/textmode.