

## Astro 复古博客重构蓝图

这个架构展示了如何将“纯 Canvas 绘图”重构为“Astro + React + Canvas 背景”的混合模式。

### 1. 布局文件 (src/layouts/Layout.astro)

这里定义网页的骨架、字体和全局变量。

```
---
// src/layouts/Layout.astro
interface Props {
  title: string;
}

const { title } = Astro.props;
---

<!doctype html>
<html lang="zh-CN">
  <head>
    <meta charset="UTF-8" />
    <meta name="viewport" content="width=device-width" />
    <link href="[https://fonts.googleapis.com/css2?family=Press+Start+2P&display="
    <title>{title}</title>
  </head>
  <body class="bg-[#222] font-pixel text-[#2f2f2f] overflow-x-hidden min-h-screen"
    <slot />
  </body>
</html>

<style is:global>
  /* 定义全局像素字体 */
  .font-pixel {
    font-family: 'Press Start 2P', cursive;
  }

  /* 定义复古滚动条 */
  ::-webkit-scrollbar {
    width: 12px;
  }
  ::-webkit-scrollbar-track {
    background: #222;
  }
  ::-webkit-scrollbar-thumb {
    background: #ff6b6b;
    border: 2px solid #2f2f2f;
  }
</style>
```

### 2. 动效背景组件 (src/components/RetroBackground.astro)

这是一个“\*\*岛”组件。我们把 Canvas 逻辑隔离在这里，只负责画背景（网格、扫描线），不再负责画按钮和文字。

```

---
// src/components/RetroBackground.astro
---

<canvas id="retro-bg" class="fixed top-0 left-0 w-full h-full -z-10 pointer-event

<script>
  // 这里放入原本 HTML 中的 Canvas 逻辑，但删掉了 drawUI 的部分
  const canvas = document.getElementById('retro-bg') as HTMLCanvasElement;
  const ctx = canvas.getContext('2d');

  let width, height, time = 0;

  function resize() {
    width = window.innerWidth;
    height = window.innerHeight;
    canvas.width = width;
    canvas.height = height;
  }

  function drawGrid() {
    ctx.strokeStyle = 'rgba(255, 255, 255, 0.05)'; // 稍微调亮一点以适应网页内容
    ctx.lineWidth = 1;
    const gridSize = 40;
    const offsetY = (time * 10) % gridSize;

    // ...这里保留原有的画线循环...
    for (let x = 0; x < width; x += gridSize) {
      ctx.beginPath(); ctx.moveTo(x, 0); ctx.lineTo(x, height); ctx.stroke();
    }
    for (let y = -gridSize; y < height; y += gridSize) {
      ctx.beginPath(); ctx.moveTo(0, y + offsetY); ctx.lineTo(width, y + offsetY);
    }
  }

  function drawScanlines() {
    ctx.fillStyle = "rgba(18, 16, 16, 0.1)";
    for (let i = 0; i < height; i += 4) {
      ctx.fillRect(0, i, width, 1);
    }
  }

  function animate() {
    requestAnimationFrame(animate);
    time += 0.05;
    ctx.clearRect(0, 0, width, height);

    // 背景色改由 CSS 控制，这里只画特效
    drawGrid();
    drawScanlines();
  }

  window.addEventListener('resize', resize);
  resize();
  animate();
</script>

```

### 3. UI 组件 (src/components/RetroCard.jsx)

这是 React 组件。注意我们不再用 Canvas 画矩形，而是用 `div` 和 CSS `box-shadow` 来模拟那个硬边框效果。

```
// src/components/RetroCard.jsx
import React from 'react';

export default function RetroCard({ title, date, desc }) {
  return (
    <div className="
      group relative
      bg-[#ffffff]
      border-4 border-[#2f2f2f]
      p-6 mb-8
      shadow-[6px_6px_0px_0px_#2f2f2f]
      transition-all duration-100
      hover:-translate-y-1 hover:shadow-[8px_8px_0px_0px_#4ecdc4]
      active:translate-y-1 active:shadow-none
      cursor-pointer
    ">
      <h3 className="text-[#ff6b6b] text-lg mb-2 leading-tight">{title}</h3>
      <div className="text-xs text-gray-500 mb-4">{date}</div>
      <p className="text-xs leading-relaxed text-[#2f2f2f]">{desc}</p>

      {/* 只有鼠标悬停时才显示的提示 */}
      <div className="hidden group-hover:block text-xs text-[#4ecdc4] text-right
        >> CLICK TO READ
      </div>
    </div>
  );
}
```

### 4. 首页组装 (src/pages/index.astro)

最后，我们在 Astro 页面里把它们组合起来。

```
---
// src/pages/index.astro
import Layout from '../layouts/Layout.astro';
import RetroBackground from '../components/RetroBackground.astro';
import RetroCard from '../components/RetroCard.jsx'; // 引入 React 组件

// 模拟数据（在真实项目中，这里会从 Markdown 文件读取）
const posts = [
  { title: "Hello World", date: "2025-10-01", desc: "Astro makes components easy." },
  { title: "Zero JS", date: "2025-10-05", desc: "Why render static HTML is better" }
];
---

<Layout title="Retro Blog">
  <!-- 1. 背景层 -->
  <RetroBackground />

  <!-- 2. 内容层 -->
```

```
<main class="relative z-10 container mx-auto px-4 py-10 max-w-2xl">

  <!-- Header -->
  <header class="text-center mb-16 border-b-4 border-[#2f2f2f] bg-[#ff6b6b] p-8">
    <h1 class="text-white text-3xl mb-2 drop-shadow-[4px_4px_0_#2f2f2f]">RETRO.
    <p class="text-[#2f2f2f] text-sm mt-4 bg-white inline-block px-2 py-1">Welc
  </header>

  <!-- Post List -->
  <div class="space-y-6">
    {posts.map(post => (
      <!-- client:visible 意味着只有当用户滚动到这个组件时，才会加载 React JS -->
      <RetroCard client:visible {...post} />
    ))}
  </div>

</main>
</Layout>
```

## 总结：Astro 带来的改变

### 1. 真正的 DOM 元素：

- 在原版 Canvas 中，文字是画上去的图片，**无法选中复制**，搜索引擎也**看不懂**。
- 在 Astro 版本中，`<h1>` 和 `<p>` 都是真正的 HTML 标签，**SEO 友好且无障碍友好**。

### 2. 性能优化：

- 背景动画（Canvas）独立运行，不会阻塞主线程。
- React 组件按需加载（`client:visible`），首屏加载极快。

### 3. 开发体验：

- 你不用再去算 `x`, `y` 坐标来对齐卡片了，直接用 CSS `margin` 和 `padding`，开发效率提升 10 倍。