

CORSO GRATUITO · 7 LEZIONI

Siti 3D con GSAP e WebGL

Il movimento fatto bene funziona ovunque. La scena vera, dentro a un canvas, si paga: qui impari quando vale il prezzo.

Federico Boggia
federicoboggia.it

I siti 3D dei premi hanno una cosa in comune: costano.

Peso da scaricare, batteria di chi guarda, ore di chi li fa. Ogni tanto valgono il prezzo.

01

Quando il 3D vale la pena

Tre mestieri con lo stesso nome, e il conto dei costi.

Federico Boggia
federicoboggia.it

Tre cose con lo stesso nome

01

Il movimento che segue lo scorrimento

Elementi piatti che si spostano mentre scendi. Lo fa GSAP, in pochi kilobyte, e funziona ovunque.

02

Un oggetto 3D nella pagina

Una scarpa che gira, un prodotto che si smonta. Serve WebGL: un modello, una luce, una camera.

03

Il sito che è una scena

Si naviga dentro a un mondo. Costa come un piccolo videogioco, e va provato su venti dispositivi.

Il corso fa le prime due. La terza serve a sapere dove finisce la strada.

70 per cento

di quello che sembra un sito 3D è movimento piatto

Niente WebGL: lo fa GSAP da solo. Negli strumenti per sviluppatori cerca un canvas, e se manca è movimento piatto.

Il 3D si guadagna il posto quando

La forma conta e va girata: una sedia, una scarpa, un macchinario

Bisogna far capire un dentro: un motore che si smonta, una casa in sezione

La configurazione è il prodotto: colore, materiale, misura

Lo spazio è il messaggio: un percorso, una mappa in rilievo

Tutto il resto è decorazione, e la decorazione si paga senza incassare.

Il conto: quattro voci

01

Il peso

Three.js compresso sta sotto i 200 KB. Il peso vero lo fanno modelli e texture.

02

Il tempo

La prima scena in un pomeriggio. Farla stare bene su un telefono di tre anni fa, un’altra settimana.

03

La batteria

La scheda grafica resta accesa: sul portatile si sente la ventola, sul telefono la mano.

04

Chi resta fuori

WebGL spento, schede in lista nera, chi ha chiesto di ridurre le animazioni.

Se togliendo il 3D la pagina regge, il 3D è un di più.

Si carica dopo, si può spegnere, e se manca nessuno se ne accorge.

02

Il movimento

GSAP: to e from, timeline, ease. Intorno ai 70 KB, ed è gratuito, plugin compresi.

GSAP in pagina, con la versione scritta

```
<script src="https://cdn.jsdelivr.net/npm/gsap@3.15.0/dist/gsap.min.js"></script>
<script src="https://cdn.jsdelivr.net/npm/gsap@3.15.0/dist/ScrollTrigger.min.js"></script>

// da npm, con un impacchettatore
import { gsap } from "gsap";
import { ScrollTrigger } from "gsap/ScrollTrigger";
gsap.registerPlugin(ScrollTrigger);
```

Con gsap@latest la versione nuova se la prende il tuo sito, senza che nessuno l'abbia provata.

to, from, fromTo: una frase sola

Questi elementi, verso questi valori, in questo tempo.

```
// da dove sta adesso, VERSO questi valori
gsap.to(".pallino", { x: 300, rotation: 180, duration: 1 });

// DA questi valori, verso dove sta adesso: e' quello che userai il 90% delle volte
gsap.from(".titolo", { y: 40, opacity: 0, duration: 0.8 });

// da qui a qui, senza guardare dove sta
gsap.fromTo(".barra", { scaleX: 0 }, { scaleX: 1, duration: 0.6 });
```

Scrivi il titolo dove deve stare, GSAP ce lo fa arrivare.

Per questo si usa from: se lo script non arriva, la pagina è già giusta. Con to resterebbe rotta.

Lo sfalsamento, la riga che rende di più

Otto centesimi di secondo di scarto, e sei card che appaiono insieme diventano sei card che arrivano.

```
gsap.from(".card", {
  y: 30,
  opacity: 0,
  duration: 0.5,
  stagger: 0.08,        // 80 millisecondi di scarto fra una card e la successiva
  ease: "power3.out"
});
```

La timeline mette in fila

Il terzo argomento è la posizione: «-=0.3» parte tre decimi prima della fine del precedente, «<» insieme, 0.5 a mezzo secondo dall'inizio.

```
const t = gsap.timeline({
  defaults: { duration: 0.6, ease: "power3.out" }   // vale per tutti, si scrive una volta
});

t.from(".pastiglia",   { y: 20, opacity: 0 })
 .from(".titolo",      { y: 40, opacity: 0 }, "-=0.3")
 .from(".sottotitolo", { y: 20, opacity: 0 }, "-=0.4")
 .from(".bottone",     { scale: 0.9, opacity: 0 }, "-=0.2");
```

L’ease, dove sta il gusto

EASE	COME SI SENTE	QUANDO
"none"	Velocità costante, meccanica	Barre di avanzamento, rotazioni continue
"power2.out"	**Parte veloce e si posa**	**Entrate, quasi sempre questa**
"power3.out"	Come sopra, più deciso	Titoli, elementi grossi
"power2.in"	Parte piano e accelera	Uscite, cose che se ne vanno
"back.out(1.7)"	Supera e torna indietro	Pulsanti, pastiglie, cose piccole
"elastic.out(1, 0.5)"	Rimbalza	Con parsimonia, e mai su un testo lungo

Chi entra usa un out, chi esce usa un in.

Fra 0,3 e 0,8 secondi sta quasi tutto: sotto i 2 decimi non si legge, sopra il secondo la gente aspetta.

Chi ha chiesto meno movimento

Emicranie, vertigini, disturbi vestibolari: chi accende «riduci le animazioni» ha un motivo serio. Fuori dal blocco, con from, la pagina è già a posto.

```
const mm = gsap.matchMedia();

mm.add("(prefers-reduced-motion: no-preference)", () => {
  // qui dentro tutte le animazioni: girano solo per chi non ha chiesto di ridurle
  gsap.from(".card", { y: 30, opacity: 0, duration: 0.5, stagger: 0.08 });
});
```

03

Lo scorrimento

ScrollTrigger mette la posizione della pagina al posto dell’orologio.

Far scattare: il primo ScrollTrigger

«top 80%»: il bordo alto della sezione all’80 per cento dello schermo.

```
gsap.from(".sezione-due .card", {
  y: 40,
  opacity: 0,
  duration: 0.6,
  stagger: 0.1,
  ease: "power3.out",
  scrollTrigger: {
    trigger: ".sezione-due",   // l'elemento da sorvegliare
    start: "top 80%",          // quando il SUO top tocca l'80% dell'altezza schermo
    toggleActions: "play none none reverse"
  }
});
```

scrub: l'animazione al guinzaglio

Sta dove sei arrivato tu, e se torni su torna indietro. Con true è incollata al dito, con 0.5 insegue con mezzo secondo di ritardo.

```
gsap.to(".barra", {
  scaleX: 1,
  ease: "none",                // con scrub l'ease va quasi sempre tolto
  scrollTrigger: {
    trigger: ".capitolo",
    start: "top top",
    end: "bottom bottom",
    scrub: 0.5                 // mezzo secondo di ritardo morbido dietro al dito
  }
});
```

Con scrub la curva ce l’ha il dito: l’ease si toglie.

Un power3.out sopra allo scorrimento fa due curve sovrapposte, e il movimento accelera dove non dovrebbe.

pin: la sezione che si ferma e racconta

```
const racconto = gsap.timeline({
  scrollTrigger: {
    trigger: ".racconto",
    start: "top top",
    end: "+=2000",     // quanto scorrimento serve per attraversarla: 2000 pixel
    scrub: 0.5,
    pin: true
  }
});

racconto
  .from(".passo-1", { opacity: 0, y: 30 })
  .to(".passo-1",   { opacity: 0, y: -30 })
  .from(".passo-2", { opacity: 0, y: 30 });
```

SCORRIMENTO PER OGNI PASSAGGIO

600 px

sotto, non si fa in tempo a leggere

2000 px

sopra, sembra che il sito si sia bloccato

Conta un migliaio di pixel per passaggio, poi aggiusta guardando.

Tre guai che capitano a tutti

1. Le misure prese troppo presto: ScrollTrigger.refresh() quando il contenuto è arrivato, e le dimensioni alle immagini
2. Il blocco sul telefono: su uno schermo alto e stretto il racconto si impila, con matchMedia sopra i 900px
3. Troppi ScrollTrigger: a ogni cambio di misura della finestra si ricalcolano tutti

E dimenticare registerPlugin: nessun errore, e lo scorrimento non fa niente.

04

La prima scena

Three.js con gli import map, i cinque pezzi e il giro di disegno.

Three.js si carica con un import map

three.min.js non esiste più. L’import map dice dove prendere «three».

```
<script type="importmap">
{
  "imports": {
    "three": "https://cdn.jsdelivr.net/npm/three@0.186.0/build/three.module.js",
    "three/addons/": "https://cdn.jsdelivr.net/npm/three@0.186.0/examples/jsm/"
  }
}
</script>
<script type="module">
  import * as THREE from 'three';
  import { OrbitControls } from 'three/addons/controls/OrbitControls.js';
</script>
```

Stessa versione
sul cuore e sugli accessori.

three@0.186.0 in tutte e due le righe. Con due versioni hai due copie di Three.js in memoria, ed errori che non dicono niente.

I cinque pezzi, più il giro

Canvas

Il rettangolo dove si disegna. La misura gliela dà il CSS.

Scena

L'elenco di quello che esiste: oggetti e luci.

Camera

Il punto di vista: angolo di campo, proporzione, vicino e lontano.

Renderer

Prende scena e camera e sporca i pixel del canvas.

Mesh

Un oggetto: la geometria, cioè la forma, più il materiale.

Il giro

Ridisegna a ogni fotogramma, perché niente si aggiorna da sé.

Forma, materiale e il giro

Con MeshStandardMaterial servono le luci: senza, vedi nero.

```
const forma = new THREE.BoxGeometry(1.5, 1.5, 1.5);
const materiale = new THREE.MeshStandardMaterial({
  color: 0x16a34a, roughness: 0.35, metalness: 0.1
});
const cubo = new THREE.Mesh(forma, materiale);
scena.add(cubo);

renderer.setAnimationLoop(() => {
  cubo.rotation.y += 0.01;
  cubo.rotation.x += 0.004;
  renderer.render(scena, camera);
});
```

IL RAPPORTO DEI PIXEL

9 volte

i pixel da disegnare, col rapporto di 3 di un telefono

2

il tetto: bordi puliti, e più di metà del lavoro in meno

Math.min(window.devicePixelRatio, 2) è la riga che più spesso trasforma una scena che scatta in una scena fluida.

05

Materiali e luci

Perché una scena appena montata sembra finta, e come smette di sembrarlo.

Perché sembra finta

01

Niente attorno

Nella tua scena c’è una lampadina nel vuoto. È la ragione principale, e si toglie con due righe.

02

Superfici perfette

Una plastica vera ha micrograffi, una vernice ha la buccia d’arancia. Serve roughness.

03

Colori dal tubo

Senza tone mapping i punti luminosi vanno a bianco netto. ACES li comprime come una pellicola.

metalness e roughness nel mondo vero

METALNESS	COSA	ROUGHNESS
0	Plastica lucida	da 0,1 a 0,25
0	Plastica opaca, gomma	da 0,6 a 0,9
0	Legno verniciato	0,4
1	Alluminio spazzolato	0,35
1	Cromo	0,05
1	Ferro arrugginito	0,8

metalness sta a 0 oppure a 1. Le mezze misure servono dove un metallo è coperto, per esempio da vernice scrostata.

Un metallo senza ambiente è nero.

Fa vedere quello che gli sta attorno. Con due luci e nient’altro, ottieni un oggetto scuro con due puntini.

L’ambiente: due righe che cambiano tutto

Una stanza finta già pronta, trasformata in luce che arriva da tutte le direzioni. Lo 0,04 è quanto sfocare i riflessi.

```
import { RoomEnvironment } from 'three/addons/environments/RoomEnvironment.js';

const pmrem = new THREE.PMREMGenerator(renderer);
scena.environment = pmrem.fromScene(new RoomEnvironment(), 0.04).texture;
```

L’ambiente fa il grosso, una direzionale dà la direzione, e basta.

sRGB solo sulle mappe di colore

Colore ed emissione sono colori. Rugosità e normali sono numeri.

```
const loader = new THREE.TextureLoader();

const colore = loader.load('/texture/legno-colore.jpg');
colore.colorSpace = THREE.SRGBColorSpace;   // SOLO sulla mappa del colore
const rugosita = loader.load('/texture/legno-roughness.jpg');
const normali  = loader.load('/texture/legno-normal.jpg');

const materiale = new THREE.MeshStandardMaterial({
  map: colore,
  roughnessMap: rugosita,
  normalMap: normali
});
```

06

Il 3D che segue il dito

GSAP comanda la scena, il canvas sta fermo, e si disegna solo quando serve.

GSAP anima anche Three.js

La rotazione di un mesh è un oggetto JavaScript con dentro tre numeri. Gli angoli sono in radianti: un giro è Math.PI * 2.

```
gsap.to(cubo.rotation, {
  y: Math.PI * 2,            // un giro intero, in radianti
  ease: "none",              // con scrub l'ease lo fa il dito
  scrollTrigger: {
    trigger: ".sezione-prodotto",
    start: "top top",
    end: "bottom bottom",
    scrub: 0.6
  }
});
```

Canvas fermo, contenuto che scorre

È l’impianto di quasi tutti i siti 3D, e si fa in CSS, senza pin: meno cose che si rompono.

```
#vista {
  position: fixed;
  inset: 0;
  width: 100%;
  height: 100%;
  z-index: 0;
}
.contenuto  { position: relative; z-index: 1; }
.contenuto section { min-height: 100vh; display: grid; place-items: center; }
```

La camera attraversa lo spazio

```
const t = gsap.timeline({
  scrollTrigger: {
    trigger: ".contenuto",
    start: "top top",
    end: "bottom bottom",
    scrub: 0.8,
    onUpdate: () => { daRidisegnare = true; }
  }
});

t.to(camera.position, { x: 0,  y: 1,  z: 3 })      // dalla panoramica al dettaglio
 .to(camera.position, { x: -2, y: 0.5, z: 2 })     // gira attorno
 .to(camera.position, { x: 0,  y: 3,  z: 6 });     // si allontana per chiudere
```

lookAt è un ordine che vale una volta.

Se muovi la camera e non lo ripeti, trasla restando girata come prima. Le inquadrature trovale con OrbitControls, e copia camera.position nella timeline.

Si ridisegna solo quando serve

Da ferma la pagina consuma quanto una pagina normale, e si accende solo mentre si scorre.

```
let daRidisegnare = true;

renderer.setAnimationLoop(() => {
  if (!daRidisegnare) return;
  camera.lookAt(0, 0, 0);
  renderer.render(scena, camera);
  daRidisegnare = false;
});

// e ogni cosa che cambia la scena alza la mano:
window.addEventListener('resize', () => { adatta(); daRidisegnare = true; });
```

Il ripiego si scrive adesso: alla fine non lo scrive nessuno.

Testi nell’HTML, un’immagine di scorta dietro al canvas, il controllo su WebGL prima di partire. Nel canvas Google non legge niente.

07

Misurare prima di pubblicare

I tre numeri, il peso, e chi resta fuori.

I tre numeri

01

Fotogrammi al secondo

Il bersaglio è 60 stabile. Un 60 che ogni tanto cade a 30 si sente peggio di un 40 costante.

02

Chiamate di disegno

renderer.info.render.calls: sotto le 100 stai tranquillo, sopra le 500 il telefono lo sente.

03

Peso scaricato

Scheda Rete, cache vuota. Sotto i 2 MB stai bene, sopra i 5 perdi gente prima della scena.

Il peso sta in due posti

IL MODELLO

glTF, nella forma .glb.

Draco e meshopt lo stringono di parecchie volte. Prima ancora, guarda quanti triangoli ha.

LE TEXTURE

KTX2 con Basis.

Un JPG nella scheda grafica sta decompresso, KTX2 resta compresso. E spesso 1024 basta, a un quarto del costo di 2048.

900.000

triangoli, per un oggetto che in pagina sarà grande 500 pixel

Nei modelli che arrivano dai reparti tecnici è la regola. Guardare quel numero rende più di qualunque compressione.

La scena si carica dopo

La pagina si apre come una pagina normale, e chi non scende fin lì non scarica niente.

```
const osservatore = new IntersectionObserver((voci) => {
  for (const v of voci) {
    if (!v.isIntersecting) continue;
    osservatore.disconnect();
    import('./scena.js').then(m => m.avvia());   // il codice arriva adesso
  }
}, { rootMargin: '200px' });                      // un po' prima di arrivarci

osservatore.observe(document.querySelector('#vista'));
```

La checklist, prima di pubblicare

La pagina si legge tutta con JavaScript disattivato

Con WebGL spento resta un’immagine sensata

Con «riduci animazioni» non si muove niente, e si capisce tutto

La scena si carica dopo, e da ferma non ridisegna

setPixelRatio limitato, markers di ScrollTrigger spenti

Provata su un telefono vero, oltre che nel simulatore



CORSO GRATUITO · 7 LEZIONI

Il verdetto può essere «non vale il prezzo».

Se ci arrivi misurando, hai la competenza che il corso cercava dalla prima lezione. Sette lezioni gratuite, ognuna con un laboratorio sul tuo progetto.

Siti 3D con GSAP e WebGL federicoboggia.it/corsi/siti-3d

Tutti i corsi gratuiti federicoboggia.it/corsi

Federico Boggia
federicoboggia.it