Proyecto Pecera Virtual — Documento de Referencia (Compilado del chat)

Este documento compila todas las decisiones y materiales que fuimos tomando en este chat. Sirve como base única para diseño, arte y prototipo. Incluye visión, estados, interacciones, tiempos tentativos, y los códigos HTML completos de los prototipos (huevo/eclosión, bebé, joven, adulto).

# 1. Visión y estilo

- Tono visual: simple y “cutrecillo simpático” (referencia POU), con estética neutra que no distrae del pez.  
- Render: Canvas 2D, sin imágenes externas (salvo que luego sustituyamos el huevo por la imagen azul nº4 agrietada).  
- Comportamiento general: animaciones suaves, nunca boca abajo, respuesta clara a interacción.

# 2. Estados del pez (arte + comportamiento)

## 2.1 Huevo

- Base visual final: usar \*\*imagen azul nº4 agrietada\*\*.  
- Eclosión: al finalizar la cuenta atrás (texto grande arriba), el huevo se \*\*desenfoca (blur)\*\* y aparece el bebé por \*\*crossfade\*\*.  
- Prototipo actual: huevo vectorial temporal con blur y transición ya funcional.

## 2.2 Bebé

- Color: lila claro.  
- Rasgos: ojos grandes; \*\*chupete\*\* en la \*\*punta\*\* de la boca, un pelín fuera de la silueta; \*\*franja vertical naranja\*\* ancha (barriga).  
- Aletas: pequeñas pero animadas.  
- Burbujeo: burbujas constantes; en turbo se multiplica la emisión.  
- Interacciones: sigue el dedo en \*\*turbo\*\* (mantener pulsado). Botón \*\*Comida\*\* (cuando esté): el bebé es \*\*muy ansioso\*\* y persigue el pellet más cercano.  
- Restricciones: no debe quedar boca abajo (limitación del ángulo).

## 2.3 Joven

- Color: violeta medio.  
- Rasgos: ojos algo más pequeños, \*\*sin chupete\*\*, franja \*\*vertical\*\* más \*\*estrecha\*\* que en bebé.  
- Aleta lateral: más \*\*alargada\*\* (evolución de forma).  
- Comida: por decisión actual, \*\*no\*\* acude a la comida (podemos ajustar más adelante).

## 2.4 Adulto

- Color: violeta oscuro.  
- Rasgos: ojos más pequeños; franja vertical \*\*muy fina\*\* (detalle). \*\*Aletas grandes y largas\*\*.  
- Movimiento: amplitud algo mayor, aleteo más pausado.  
- Comida: por ahora no acude.

# 3. Interacciones y controles

- Mantener pulsado: \*\*TURBO\*\* → los peces siguen el dedo y sueltan muchas más burbujas; vuelve a normal al soltar.  
- Botón \*\*Comida\*\*: caen pellets desde arriba; \*\*solo el bebé\*\* se lanza ansioso a comer.  
- Botón \*\*Reset\*\*: recoloca/reinicia estado.  
- (Prototipo multipez) \*\*Intercambiar lados\*\*: cambia posiciones iniciales.  
- Opción de \*\*contorno\*\* de debug (en una versión) para verificar visibilidad.

# 4. Eclosión y naming

- Cuenta atrás grande arriba: “Quedan X min/seg para que nazca tu pececillo”.  
- Al llegar a 0: blur del huevo → crossfade al bebé.  
- \*\*Burbuja de nombre\*\*: tras nacer, aparece una burbuja flotante que el jugador toca para escribir el nombre del pez.

# 5. Tiempos / evolución (tentativos)

Estos tiempos se configurarán en la app; guardamos aquí la referencia conversada:  
- Huevo: ~2 días (idea inicial mencionada).  
- Bebé: ~5–7 días (muy mono, se mantiene más tiempo).  
- Joven: a partir del ~día 7.  
- Adulto: tras la etapa de joven (a definir exacto en backoffice).  
Nota: estos valores son orientativos y se parametrizarán.

# 6. Burbujas y orientación

- Fondo: burbujas ambientales constantes.  
- Boca: emisión periódica; \*\*en turbo\*\* aumenta la frecuencia y a veces dobles/triples burbujas.  
- Colisiones y márgenes: rebote suave; \*\*nunca se orienta boca abajo\*\* (ángulo limitado).

# 7. Arte de pecera y futuros

- Pecera neutra actual; pendiente: plantas, piedras, decoraciones.  
- Estados especiales: enfermo, dormido, contento.  
- Limpieza de pecera u otras acciones de cuidado.

# 8. Prototipos y archivos generados

- Huevo / Eclosión (HTML): huevo\_eclosion.html

- Bebé (franja vertical, chupete): pecera\_bebe\_franja\_vertical.html

- Joven (franja vertical, aleta alargada): pez\_joven.html

- Adulto (franja vertical fina, aletas grandes): pez\_adulto.html

# 9. Decisiones y cambios a lo largo del chat

- \*\*Chupete\*\*: inicialmente quedaba atrasado; decisión final → en la \*\*punta\*\* de la boca y un poco fuera de la silueta.  
- \*\*Franja\*\*: empezó horizontal (cinturón de cabeza a cola); decisión final → \*\*vertical\*\* (barriga) y \*\*se estrecha con la edad\*\*.  
- \*\*Visibilidad\*\*: hubo versiones donde el pez no se veía; se añadieron contornos, contraste y versiones mínimas para verificación.  
- \*\*Comida\*\*: solo el bebé se lanza; joven/adulto pasan (ajustable más adelante).

# Anexos — Código fuente de los prototipos

## Huevo / Eclosión (HTML)

|  |
| --- |
| <!doctype html>  <html lang="es">  <head>  <meta charset="utf-8" />  <meta name="viewport" content="width=device-width,initial-scale=1,maximum-scale=1,user-scalable=no"/>  <title>Huevo → Eclosión (blur) → Bebé</title>  <style>  :root { --bg:#08243a; --text:#e9f6ff; --ui:#1d4b6b; --ui2:#2a6287; }  html,body{height:100%;margin:0;background:var(--bg);color:var(--text);font-family:system-ui,Segoe UI,Roboto,Arial,sans-serif;overflow:hidden;}  #topbar{position:fixed;left:12px;right:12px;top:10px;display:flex;align-items:center;justify-content:space-between;gap:10px}  #count{font-weight:800;letter-spacing:.5px}  #btn{appearance:none;border:0;border-radius:12px;padding:10px 14px;background:var(--ui);color:var(--text);font-weight:700;box-shadow:0 6px 20px rgba(0,0,0,.25);}  #btn:active{transform:translateY(1px);background:var(--ui2)}  .stage{position:fixed;inset:0;display:block;opacity:1;transition:opacity .45s ease}  .hidden{opacity:0;pointer-events:none}  /\* Blur animado para el huevo \*/  .blur-in{filter:blur(0px);transform:scale(1);transition:filter .6s ease, transform .6s ease; }  .blurred{filter:blur(10px);transform:scale(1.04);}  /\* Badge TURBO como antes por consistencia cuando aparece el bebé \*/  #turbo{position:fixed;right:12px;top:50px;padding:6px 10px;border-radius:999px;background:rgba(255,255,255,.12);backdrop-filter:blur(4px);font-weight:700;opacity:0;transition:opacity .15s}  #turbo.on{opacity:1}  /\* Pista visual de cambio \*/  #toast{position:fixed;left:50%;top:50%;transform:translate(-50%,-50%);padding:10px 14px;border-radius:12px;background:rgba(0,0,0,.35);backdrop-filter:blur(2px);opacity:0;transition:opacity .3s}  #toast.show{opacity:1}  </style>  </head>  <body>  <div id="topbar">  <div id="count">Eclosión en: <span id="t">5</span>s</div>  <button id="btn">Eclosionar ahora</button>  </div>  <div id="turbo">TURBO 💨</div>  <!-- Escena del huevo -->  <canvas id="egg" class="stage blur-in"></canvas>  <!-- Escena del bebé -->  <canvas id="baby" class="stage hidden"></canvas>  <div id="toast">¡Nace el pececito! 🐣➡️🐟</div>  <script>  (()=>{  const egg = document.getElementById('egg');  const baby = document.getElementById('baby');  const ctxE = egg.getContext('2d', { alpha:false });  const ctxB = baby.getContext('2d', { alpha:false });  function resize(){ egg.width=innerWidth; egg.height=innerHeight; baby.width=innerWidth; baby.height=innerHeight; }  addEventListener('resize', resize, {passive:true}); resize();  // ----- Escena HUEVO (estático con flotación suave) -----  const hue = { x: innerWidth\*0.5, y: innerHeight\*0.55, t: 0 };  function drawEgg(dt){  hue.t += dt;  const y = hue.y + Math.sin(hue.t\*0.8)\*6;  const w = 110, h = 140;  // fondo  const g = ctxE.createLinearGradient(0,0,0,egg.height);  g.addColorStop(0,'#0b304d'); g.addColorStop(1,'#061726');  ctxE.fillStyle = g; ctxE.fillRect(0,0,egg.width,egg.height);  // burbujitas de fondo  ctxE.globalAlpha = 0.28;  ctxE.fillStyle = '#cfe9ff';  for(let i=0;i<18;i++){  const bx = (i\*97 % egg.width);  const by = (egg.height - ((hue.t\*30 + i\*120)% (egg.height+60)));  const rr = 1 + (i%3);  ctxE.beginPath(); ctxE.arc(bx,by,rr,0,Math.PI\*2); ctxE.fill();  }  ctxE.globalAlpha = 1;  // huevo  ctxE.save();  ctxE.translate(hue.x,y);  ctxE.fillStyle = '#f4f2e7';  ctxE.strokeStyle = 'rgba(40,50,60,.25)';  ctxE.lineWidth = 3;  ctxE.beginPath();  ctxE.ellipse(0,0,w\*0.62,h\*0.72,0,0,Math.PI\*2);  ctxE.fill(); ctxE.stroke();  // sombra/volumen  ctxE.globalAlpha = .25;  ctxE.fillStyle = '#c8c2b0';  ctxE.beginPath();  ctxE.ellipse(-15,-20,w\*0.28,h\*0.22,0,0,Math.PI\*2);  ctxE.fill();  ctxE.globalAlpha = 1;  // base/arena simple  ctxE.fillStyle = '#16324a';  ctxE.beginPath();  ctxE.ellipse(0, h\*0.48, w\*0.8, h\*0.18, 0, 0, Math.PI\*2);  ctxE.fill();  ctxE.restore();  }  // ----- Transición: BLUR y crossfade -----  let hatched = false;  function hatch(){  if (hatched) return;  hatched = true;  // blur al huevo  egg.classList.add('blurred');  // pequeño toast  const toast = document.getElementById('toast');  toast.classList.add('show');  setTimeout(()=>toast.classList.remove('show'), 1200);  // crossfade a bebé  setTimeout(()=>{  egg.classList.add('hidden');  baby.classList.remove('hidden');  }, 600); // coincide con la transición de blur  }  // Cuenta atrás  let T = 5;  const tEl = document.getElementById('t');  const timer = setInterval(()=>{  if (hatched) { clearInterval(timer); return; }  T--; tEl.textContent = T;  if (T<=0){ clearInterval(timer); hatch(); }  }, 1000);  document.getElementById('btn').onclick = hatch;  // ----- Escena BEBÉ (canvas reutilizando estilo) -----  const turboBadge = document.getElementById('turbo');  const rand=(a,b)=>a+Math.random()\*(b-a);  const lerp=(a,b,t)=>a+(b-a)\*t;  const clampAngle=(a)=>{const m=70\*Math.PI/180;return Math.max(-m,Math.min(m,a));};  const BABY={ size:46, body:'#d9ccff', tail:'#ff9ecf', fin:'#ffd0e2', eyeScale:1.6, stripeColor:'#ffba7a', stripeWidth:0.32 };  let fish={ x: baby.width\*0.5, y: baby.height\*0.55, vx: 110, vy: 8, baseSpeed: 120, followSpeed: 260, wobbleT: Math.random()\*10, angle: 0, target: null, bubbleTimer: 0.4, turbo:false };  const mouthBubbles=[]; const bgBubbles=[];  for(let i=0;i<24;i++){ bgBubbles.push({x:Math.random()\*baby.width,y:baby.height+Math.random()\*baby.height,r:1+Math.random()\*3,s:12+Math.random()\*24}); }  let pointer=null;  baby.addEventListener('pointerdown',e=>{ pointer={x:e.clientX,y:e.clientY}; fish.target=pointer; fish.turbo=true; turboBadge.classList.add('on'); });  baby.addEventListener('pointermove',e=>{ if(pointer){ pointer.x=e.clientX; pointer.y=e.clientY; } });  baby.addEventListener('pointerup',()=>{ pointer=null; fish.target=null; fish.turbo=false; turboBadge.classList.remove('on'); });  baby.addEventListener('pointercancel',()=>{ pointer=null; fish.target=null; fish.turbo=false; turboBadge.classList.remove('on'); });  function addMouthBubble(turboMul){  const dir=Math.sign(fish.vx)||1;  const mouthX=fish.x+(dir>0?BABY.size\*0.70:-BABY.size\*0.70);  mouthBubbles.push({x:mouthX,y:fish.y,v:rand(22,34)\*(turboMul?1.25:1),r:rand(1.2,2.4)\*(turboMul?1.15:1),life:rand(0.9,1.4)\*(turboMul?1.2:1)});  }  function drawStripeVertical(rx,ry,color,widthRatio){  const halfW=rx\*widthRatio;  const margin=2;  const rxIn=rx-margin, ryIn=ry-margin;  ctxB.save(); ctxB.fillStyle=color; ctxB.globalAlpha=0.95; ctxB.beginPath();  const steps=48;  for(let i=0;i<=steps;i++){  const t=i/steps;  const x=-halfW + t\*(2\*halfW);  const y=ryIn\*Math.sqrt(Math.max(0,1-(x\*x)/(rxIn\*rxIn)));  if(i===0) ctxB.moveTo(x,-y); else ctxB.lineTo(x,-y);  }  for(let i=steps;i>=0;i--){  const t=i/steps;  const x=-halfW + t\*(2\*halfW);  const y=ryIn\*Math.sqrt(Math.max(0,1-(x\*x)/(rxIn\*rxIn)));  ctxB.lineTo(x,y);  }  ctxB.closePath(); ctxB.fill(); ctxB.globalAlpha=1; ctxB.restore();  }  function drawBaby(time,dt){  // fondo  const g = ctxB.createLinearGradient(0,0,0,baby.height);  g.addColorStop(0,'#0b304d'); g.addColorStop(1,'#061726');  ctxB.fillStyle = g; ctxB.fillRect(0,0,baby.width,baby.height);  // burbujas fondo  ctxB.globalAlpha=0.35; ctxB.fillStyle='#cfe9ff';  for(const b of bgBubbles){ b.y-=b.s\*dt; if(b.y<-10){ b.y=baby.height+10; b.x=Math.random()\*baby.width; } ctxB.beginPath(); ctxB.arc(b.x,b.y,b.r,0,Math.PI\*2); ctxB.fill(); }  ctxB.globalAlpha=1;  // avance del pez  const pad=70;  let desiredVx=(Math.sign(fish.vx)||1)\*fish.baseSpeed;  let desiredVy=fish.vy;  if (fish.target){ const dx=fish.target.x-fish.x, dy=fish.target.y-fish.y; const len=Math.hypot(dx,dy)||1; const spd=fish.followSpeed\*1.6; desiredVx=(dx/len)\*spd; desiredVy=(dy/len)\*spd; }  else { fish.wobbleT+=dt; desiredVy=fish.vy+Math.sin(fish.wobbleT\*1.6)\*10\*dt; }  fish.bubbleTimer -= dt \* (fish.target?2.6:1.0);  if (fish.bubbleTimer<=0){ addMouthBubble(!!fish.target); if (fish.target){ if(Math.random()<0.8)addMouthBubble(true); if(Math.random()<0.55)addMouthBubble(true);} fish.bubbleTimer=0.45+Math.random()\*0.65; }  fish.vx = lerp(fish.vx,desiredVx,Math.min(1,dt\*3.0));  fish.vy = lerp(fish.vy,desiredVy,Math.min(1,dt\*3.0));  fish.x += fish.vx\*dt; fish.y += fish.vy\*dt;  if (fish.x<pad){fish.x=pad;fish.vx=Math.abs(fish.vx);} if(fish.x>baby.width-pad){fish.x=baby.width-pad;fish.vx=-Math.abs(fish.vx);}  if (fish.y<pad\*0.7){fish.y=pad\*0.7;fish.vy=Math.abs(fish.vy);} if(fish.y>baby.height-pad\*0.7){fish.y=baby.height-pad\*0.7;fish.vy=-Math.abs(fish.vy);}  fish.angle = clampAngle(Math.atan2(fish.vy, Math.abs(fish.vx)));  // dibujar pez  const s=BABY.size, flip=fish.vx<0?-1:1;  ctxB.save(); ctxB.translate(fish.x,fish.y); ctxB.scale(flip,1); ctxB.rotate(clampAngle(fish.angle));  const rx=s\*1.05, ry=s\*0.78;  // cuerpo  ctxB.fillStyle=BABY.body; ctxB.strokeStyle='rgba(0,0,0,.5)'; ctxB.lineWidth=3;  ctxB.beginPath(); ctxB.ellipse(0,0,rx,ry,0,0,Math.PI\*2); ctxB.fill(); ctxB.stroke();  // franja vertical (ancha en bebé)  drawStripeVertical(rx,ry,BABY.stripeColor,BABY.stripeWidth);  // cola  const flap=Math.sin(time\*6 + fish.wobbleT)\*0.38;  ctxB.save(); ctxB.translate(-rx,0); ctxB.rotate(flap);  ctxB.fillStyle=BABY.tail;  ctxB.beginPath(); ctxB.moveTo(0,0);  ctxB.quadraticCurveTo(-s\*0.48,-s\*0.58, s\*0.12,-s\*0.22);  ctxB.quadraticCurveTo(-s\*0.48, s\*0.58, 0, 0); ctxB.fill();  ctxB.restore();  // aleta  const finFlap=Math.sin(time\*8.5 + fish.wobbleT)\*0.27;  ctxB.save(); ctxB.translate(-s\*0.1, s\*0.16); ctxB.rotate(finFlap);  ctxB.fillStyle=BABY.fin;  ctxB.beginPath();  ctxB.moveTo(0,0);  ctxB.quadraticCurveTo( s\*0.06, s\*0.58, s\*0.2, s\*0.13);  ctxB.quadraticCurveTo( s\*0.06, s\*0.25, 0, 0); ctxB.fill();  ctxB.restore();  // ojo  const e=0.22\*BABY.eyeScale, p=0.11\*BABY.eyeScale;  ctxB.fillStyle='#fff'; ctxB.beginPath(); ctxB.arc(s\*0.46, 0, s\*e, 0, Math.PI\*2); ctxB.fill();  ctxB.fillStyle='#111'; ctxB.beginPath(); ctxB.arc(s\*0.53, 0, s\*p, 0, Math.PI\*2); ctxB.fill();  // chupete (en la punta, un pelín fuera)  const mouthTipX=rx + s\*0.06, mouthTipY=0;  ctxB.strokeStyle='#ff78c2'; ctxB.lineWidth=3.2;  ctxB.beginPath(); ctxB.arc(mouthTipX, mouthTipY, s\*0.18, 0, Math.PI\*2); ctxB.stroke();  ctxB.fillStyle='#ffb5de';  ctxB.beginPath(); ctxB.ellipse(mouthTipX - s\*0.09, mouthTipY, s\*0.17, s\*0.13, 0, 0, Math.PI\*2); ctxB.fill();  ctxB.restore();  // burbujas de la boca  for (let i=mouthBubbles.length-1;i>=0;i--){  const m=mouthBubbles[i]; m.y-=m.v\*dt; m.life-=dt;  if (m.life<=0){ mouthBubbles.splice(i,1); continue; }  ctxB.globalAlpha=Math.max(0,Math.min(1,m.life));  ctxB.strokeStyle='rgba(200,220,255,.9)'; ctxB.lineWidth=1.6;  ctxB.beginPath(); ctxB.arc(m.x,m.y,m.r,0,Math.PI\*2); ctxB.stroke();  ctxB.globalAlpha=1;  }  }  let last=performance.now();  function loop(t){  const dt = Math.min(0.033,(t-last)/1000); last = t;  // Render según etapa  if (!hatched){ drawEgg(dt); } else { drawBaby(t/1000, dt); }  requestAnimationFrame(loop);  }  requestAnimationFrame(loop);  })();  </script>  </body>  </html> |

## Bebé (franja vertical, chupete)

|  |
| --- |
| <!doctype html>  <html lang="es">  <head>  <meta charset="utf-8" />  <meta name="viewport" content="width=device-width,initial-scale=1,maximum-scale=1,user-scalable=no"/>  <title>Bebé — franja VERTICAL</title>  <style>  :root { --bg:#08243a; --ui:#1d4b6b; --ui2:#2a6287; --text:#e9f6ff;}  html,body{height:100%;margin:0;background:var(--bg);}  canvas{position:fixed;inset:0;touch-action:none}  #turbo{position:fixed;right:12px;top:12px;padding:6px 10px;border-radius:999px;  background:rgba(255,255,255,.12);backdrop-filter:blur(4px);font-weight:700;opacity:0;transition:opacity .15s}  #turbo.on{opacity:1}  </style>  </head>  <body>  <div id="turbo">TURBO 💨</div>  <canvas id="tank"></canvas>  <script>  (()=>{  const canvas=document.getElementById('tank');  const ctx=canvas.getContext('2d',{alpha:false});  function resize(){canvas.width=innerWidth;canvas.height=innerHeight;}resize();  addEventListener('resize',resize);  const turboBadge=document.getElementById('turbo');  const rand=(a,b)=>a+Math.random()\*(b-a);  const lerp=(a,b,t)=>a+(b-a)\*t;  const clampAngle=(a)=>{const m=70\*Math.PI/180;return Math.max(-m,Math.min(m,a));};  const bgBubbles=Array.from({length:24},()=>({x:Math.random()\*innerWidth,y:innerHeight+Math.random()\*innerHeight,r:1+Math.random()\*3,s:12+Math.random()\*24}));  const mouthBubbles=[];  const BABY={size:46,body:'#d9ccff',tail:'#ff9ecf',fin:'#ffd0e2',eyeScale:1.6,stripeColor:'#ffba7a',stripeWidth:0.32};  let fish={x:canvas.width\*0.5,y:canvas.height\*0.55,vx:rand(80,120),vy:rand(-20,20),baseSpeed:120,followSpeed:260,wobbleT:Math.random()\*10,angle:0,target:null,bubbleTimer:rand(0.2,0.8),turbo:false};  let pointer=null;  canvas.addEventListener('pointerdown',e=>{pointer={x:e.clientX,y:e.clientY};fish.target=pointer;fish.turbo=true;turboBadge.classList.add('on');});  canvas.addEventListener('pointermove',e=>{if(pointer){pointer.x=e.clientX;pointer.y=e.clientY;}});  canvas.addEventListener('pointerup',()=>{pointer=null;fish.target=null;fish.turbo=false;turboBadge.classList.remove('on');});  canvas.addEventListener('pointercancel',()=>{pointer=null;fish.target=null;fish.turbo=false;turboBadge.classList.remove('on');});  function addMouthBubble(turboMul){  const dir=Math.sign(fish.vx)||1;  const mouthX=fish.x+(dir>0?BABY.size\*0.70:-BABY.size\*0.70);  mouthBubbles.push({x:mouthX,y:fish.y,v:rand(22,34)\*(turboMul?1.25:1),r:rand(1.2,2.4)\*(turboMul?1.15:1),life:rand(0.9,1.4)\*(turboMul?1.2:1)});  }  function drawStripeVertical(rx,ry,color,widthRatio){  const halfW=rx\*widthRatio;  const margin=2;  const rxIn=rx-margin,ryIn=ry-margin;  ctx.save();ctx.fillStyle=color;ctx.globalAlpha=0.95;ctx.beginPath();  const steps=48;  for(let i=0;i<=steps;i++){  const t=i/steps;  const x=-halfW+t\*(2\*halfW);  const y=ryIn\*Math.sqrt(Math.max(0,1-(x\*x)/(rxIn\*rxIn)));  if(i===0)ctx.moveTo(x,-y);else ctx.lineTo(x,-y);  }  for(let i=steps;i>=0;i--){  const t=i/steps;  const x=-halfW+t\*(2\*halfW);  const y=ryIn\*Math.sqrt(Math.max(0,1-(x\*x)/(rxIn\*rxIn)));  ctx.lineTo(x,y);  }  ctx.closePath();ctx.fill();ctx.restore();  }  function drawFish(time){  const s=BABY.size,flip=fish.vx<0?-1:1;  ctx.save();ctx.translate(fish.x,fish.y);ctx.scale(flip,1);ctx.rotate(clampAngle(fish.angle));  const rx=s\*1.05,ry=s\*0.78;  ctx.fillStyle=BABY.body;ctx.strokeStyle='rgba(0,0,0,.5)';ctx.lineWidth=3;  ctx.beginPath();ctx.ellipse(0,0,rx,ry,0,0,Math.PI\*2);ctx.fill();ctx.stroke();  // franja VERTICAL barriga  drawStripeVertical(rx,ry,BABY.stripeColor,BABY.stripeWidth);  // cola  const flap=Math.sin(time\*6+fish.wobbleT)\*0.38;  ctx.save();ctx.translate(-rx,0);ctx.rotate(flap);  ctx.fillStyle=BABY.tail;ctx.beginPath();ctx.moveTo(0,0);  ctx.quadraticCurveTo(-s\*0.48,-s\*0.58,s\*0.12,-s\*0.22);  ctx.quadraticCurveTo(-s\*0.48,s\*0.58,0,0);ctx.fill();ctx.restore();  // aleta  const finFlap=Math.sin(time\*8.5+fish.wobbleT)\*0.27;  ctx.save();ctx.translate(-s\*0.1,s\*0.16);ctx.rotate(finFlap);  ctx.fillStyle=BABY.fin;ctx.beginPath();ctx.moveTo(0,0);  ctx.quadraticCurveTo(s\*0.06,s\*0.58,s\*0.2,s\*0.13);  ctx.quadraticCurveTo(s\*0.06,s\*0.25,0,0);ctx.fill();ctx.restore();  // ojo  const e=0.22\*BABY.eyeScale,p=0.11\*BABY.eyeScale;  ctx.fillStyle='#fff';ctx.beginPath();ctx.arc(s\*0.46,0,s\*e,0,Math.PI\*2);ctx.fill();  ctx.fillStyle='#111';ctx.beginPath();ctx.arc(s\*0.53,0,s\*p,0,Math.PI\*2);ctx.fill();  // chupete borde  const mouthTipX=rx+s\*0.06,mouthTipY=0;  ctx.strokeStyle='#ff78c2';ctx.lineWidth=3.2;  ctx.beginPath();ctx.arc(mouthTipX,mouthTipY,s\*0.18,0,Math.PI\*2);ctx.stroke();  ctx.fillStyle='#ffb5de';ctx.beginPath();  ctx.ellipse(mouthTipX-s\*0.09,mouthTipY,s\*0.17,s\*0.13,0,0,Math.PI\*2);ctx.fill();  ctx.restore();  }  let last=performance.now();  function tick(t){  const dt=Math.min(0.033,(t-last)/1000);last=t;  ctx.fillStyle='#08243a';ctx.fillRect(0,0,canvas.width,canvas.height);  ctx.globalAlpha=0.35;ctx.fillStyle='#cfe9ff';  for(const b of bgBubbles){b.y-=b.s\*dt;if(b.y<-10){b.y=canvas.height+10;b.x=Math.random()\*canvas.width;}ctx.beginPath();ctx.arc(b.x,b.y,b.r,0,Math.PI\*2);ctx.fill();}  ctx.globalAlpha=1;  for(let i=mouthBubbles.length-1;i>=0;i--){const m=mouthBubbles[i];m.y-=m.v\*dt;m.life-=dt;if(m.life<=0){mouthBubbles.splice(i,1);continue;}ctx.globalAlpha=Math.max(0,Math.min(1,m.life));ctx.strokeStyle='rgba(200,220,255,.9)';ctx.lineWidth=1.6;ctx.beginPath();ctx.arc(m.x,m.y,m.r,0,Math.PI\*2);ctx.stroke();ctx.globalAlpha=1;}  const pad=70;let desiredVx=(Math.sign(fish.vx)||1)\*fish.baseSpeed,desiredVy=fish.vy;  if(fish.target){const dx=fish.target.x-fish.x,dy=fish.target.y-fish.y;const len=Math.hypot(dx,dy)||1;const spd=fish.followSpeed\*1.6;desiredVx=(dx/len)\*spd;desiredVy=(dy/len)\*spd;}  else{fish.wobbleT+=dt;desiredVy=fish.vy+Math.sin(fish.wobbleT\*1.6)\*10\*dt;}  fish.bubbleTimer-=dt\*(fish.target?2.6:1.0);if(fish.bubbleTimer<=0){addMouthBubble(!!fish.target);if(fish.target){if(Math.random()<0.8)addMouthBubble(true);if(Math.random()<0.55)addMouthBubble(true);}fish.bubbleTimer=rand(0.45,1.1);}  fish.vx=lerp(fish.vx,desiredVx,Math.min(1,dt\*3.0));fish.vy=lerp(fish.vy,desiredVy,Math.min(1,dt\*3.0));  fish.x+=fish.vx\*dt;fish.y+=fish.vy\*dt;  if(fish.x<pad){fish.x=pad;fish.vx=Math.abs(fish.vx);}if(fish.x>canvas.width-pad){fish.x=canvas.width-pad;fish.vx=-Math.abs(fish.vx);}  if(fish.y<pad\*0.7){fish.y=pad\*0.7;fish.vy=Math.abs(fish.vy);}if(fish.y>canvas.height-pad\*0.7){fish.y=canvas.height-pad\*0.7;fish.vy=-Math.abs(fish.vy);}  fish.angle=clampAngle(Math.atan2(fish.vy,Math.abs(fish.vx)));  drawFish(performance.now()/1000);  requestAnimationFrame(tick);  }  requestAnimationFrame(tick);  })();  </script>  </body>  </html> |

## Joven (franja vertical, aleta alargada)

|  |
| --- |
| <!doctype html>  <html lang="es">  <head>  <meta charset="utf-8" />  <meta name="viewport" content="width=device-width,initial-scale=1,maximum-scale=1,user-scalable=no"/>  <title>Joven — franja vertical + aleta alargada</title>  <style>  :root { --bg:#08243a; --ui:#1d4b6b; --ui2:#2a6287; --text:#e9f6ff;}  html,body{height:100%;margin:0;background:var(--bg);}  body{font-family:system-ui,Segoe UI,Roboto,Helvetica,Arial,sans-serif;color:var(--text);overflow:hidden;}  canvas{position:fixed;inset:0;touch-action:none}  #bar{position:fixed;left:0;right:0;bottom:0;padding:10px;  display:flex;gap:10px;justify-content:center;flex-wrap:wrap;  background:linear-gradient(to top, rgba(0,0,0,.35), rgba(0,0,0,0));}  .btn{appearance:none;border:0;border-radius:12px;padding:10px 14px;  background:var(--ui);color:var(--text);font-weight:600;cursor:pointer;  box-shadow:0 6px 20px rgba(0,0,0,.25);display:inline-flex;gap:8px;align-items:center}  .btn:active{transform:translateY(1px);background:var(--ui2)}  #turbo{position:fixed;right:12px;top:12px;padding:6px 10px;border-radius:999px;  background:rgba(255,255,255,.12);backdrop-filter:blur(4px);font-weight:700;opacity:0;transition:opacity .15s}  #turbo.on{opacity:1}  </style>  </head>  <body>  <div id="turbo">TURBO 💨</div>  <canvas id="tank"></canvas>  <div id="bar">  <button class="btn" id="btnReset">↺ Reset</button>  </div>  <script>  (()=>{  const canvas=document.getElementById('tank');  const ctx=canvas.getContext('2d',{alpha:false});  function resize(){canvas.width=innerWidth;canvas.height=innerHeight;}resize();  addEventListener('resize',resize);  const turboBadge=document.getElementById('turbo');  const rand=(a,b)=>a+Math.random()\*(b-a);  const lerp=(a,b,t)=>a+(b-a)\*t;  const clampAngle=(a)=>{const m=70\*Math.PI/180;return Math.max(-m,Math.min(m,a));};  // Burbujas de fondo  const bgBubbles=Array.from({length:26},()=>({x:Math.random()\*innerWidth,y:innerHeight+Math.random()\*innerHeight,r:1+Math.random()\*3,s:12+Math.random()\*24}));  const mouthBubbles=[];  // Config del JOVEN (un poco más grande, violeta medio, ojo más pequeño, aleta alargada)  const JOVEN={  size:52,  body:'#b1a1ff', // violeta medio  tail:'#f191cd',  fin:'#f1c4e4',  eyeScale:1.25, // más pequeño que bebé  stripeColor:'#ffa960',  stripeWidth:0.20 // más estrecha que el bebé  };  let fish={  x:canvas.width\*0.5,  y:canvas.height\*0.55,  vx:rand(90,130),  vy:rand(-24,24),  baseSpeed:130,  followSpeed:270,  wobbleT:Math.random()\*10,  angle:0,  target:null,  bubbleTimer:rand(0.25,0.9),  turbo:false  };  // Interacción: mantener pulsado = turbo + seguir dedo  let pointer=null;  canvas.addEventListener('pointerdown',e=>{pointer={x:e.clientX,y:e.clientY};fish.target=pointer;fish.turbo=true;turboBadge.classList.add('on');});  canvas.addEventListener('pointermove',e=>{if(pointer){pointer.x=e.clientX;pointer.y=e.clientY;}});  canvas.addEventListener('pointerup',()=>{pointer=null;fish.target=null;fish.turbo=false;turboBadge.classList.remove('on');});  canvas.addEventListener('pointercancel',()=>{pointer=null;fish.target=null;fish.turbo=false;turboBadge.classList.remove('on');});  document.getElementById('btnReset').onclick=()=>{  fish.x=canvas.width\*0.5; fish.y=canvas.height\*0.55;  fish.vx=rand(90,130); fish.vy=rand(-24,24);  fish.target=null; fish.turbo=false; turboBadge.classList.remove('on');  mouthBubbles.length=0;  };  function addMouthBubble(turboMul){  const dir=Math.sign(fish.vx)||1;  const mouthX=fish.x+(dir>0?JOVEN.size\*0.70:-JOVEN.size\*0.70);  mouthBubbles.push({  x:mouthX + rand(-1.0,1.0),  y:fish.y + rand(-1.0,1.0),  v:rand(22,36)\*(turboMul?1.25:1),  r:rand(1.2,2.6)\*(turboMul?1.15:1),  life:rand(0.9,1.5)\*(turboMul?1.2:1)  });  }  function drawStripeVertical(rx,ry,color,widthRatio){  // Banda vertical ajustada al óvalo (desde lomo a barriga), no se sale  const halfW=rx\*widthRatio;  const margin=2;  const rxIn=rx-margin, ryIn=ry-margin;  ctx.save(); ctx.fillStyle=color; ctx.globalAlpha=0.95; ctx.beginPath();  const steps=56;  for(let i=0;i<=steps;i++){  const t=i/steps;  const x=-halfW + t\*(2\*halfW);  const y=ryIn\*Math.sqrt(Math.max(0,1-(x\*x)/(rxIn\*rxIn)));  if(i===0) ctx.moveTo(x,-y); else ctx.lineTo(x,-y);  }  for(let i=steps;i>=0;i--){  const t=i/steps;  const x=-halfW + t\*(2\*halfW);  const y=ryIn\*Math.sqrt(Math.max(0,1-(x\*x)/(rxIn\*rxIn)));  ctx.lineTo(x,y);  }  ctx.closePath(); ctx.fill(); ctx.globalAlpha=1; ctx.restore();  }  function drawFish(time){  const s=JOVEN.size, flip=fish.vx<0?-1:1;  ctx.save(); ctx.translate(fish.x,fish.y); ctx.scale(flip,1); ctx.rotate(clampAngle(fish.angle));  // Cuerpo ovalado con contorno  const rx=s\*1.05, ry=s\*0.78;  ctx.fillStyle=JOVEN.body;  ctx.strokeStyle='rgba(0,0,0,.5)';  ctx.lineWidth=3;  ctx.beginPath(); ctx.ellipse(0,0,rx,ry,0,0,Math.PI\*2); ctx.fill(); ctx.stroke();  // Franja vertical (más estrecha)  drawStripeVertical(rx,ry,JOVEN.stripeColor,JOVEN.stripeWidth);  // Cola (aleteo)  const flap=Math.sin(time\*6 + fish.wobbleT)\*0.36;  ctx.save(); ctx.translate(-rx,0); ctx.rotate(flap);  ctx.fillStyle=JOVEN.tail;  ctx.beginPath();  ctx.moveTo(0,0);  ctx.quadraticCurveTo(-s\*0.5, -s\*0.6, s\*0.14, -s\*0.24);  ctx.quadraticCurveTo(-s\*0.5, s\*0.6, 0, 0);  ctx.fill();  ctx.restore();  // Aleta lateral ALARGADA  const finFlap=Math.sin(time\*8.2 + fish.wobbleT)\*0.22;  ctx.save();  ctx.translate(-s\*0.05, s\*0.12); // un poco más arriba y adelantada  ctx.rotate(finFlap);  ctx.fillStyle=JOVEN.fin;  ctx.beginPath();  // Alargar: mayor extensión hacia abajo y adelante  ctx.moveTo(0,0);  ctx.quadraticCurveTo( s\*0.10, s\*0.80, s\*0.28, s\*0.22);  ctx.quadraticCurveTo( s\*0.08, s\*0.30, 0, 0);  ctx.fill();  ctx.restore();  // Ojo (más pequeño que bebé)  const e=0.22 \* JOVEN.eyeScale, p=0.11 \* JOVEN.eyeScale;  ctx.fillStyle='#fff'; ctx.beginPath(); ctx.arc(s\*0.46, 0, s\*e, 0, Math.PI\*2); ctx.fill();  ctx.fillStyle='#111'; ctx.beginPath(); ctx.arc(s\*0.53, 0, s\*p, 0, Math.PI\*2); ctx.fill();  ctx.restore();  }  let last=performance.now();  function tick(t){  const dt=Math.min(0.033,(t-last)/1000); last=t;  // Fondo + burbujas ambientales  ctx.fillStyle='#08243a'; ctx.fillRect(0,0,canvas.width,canvas.height);  ctx.globalAlpha=0.35; ctx.fillStyle='#cfe9ff';  for(const b of bgBubbles){  b.y-=b.s\*dt;  if(b.y<-10){ b.y=canvas.height+10; b.x=Math.random()\*canvas.width; }  ctx.beginPath(); ctx.arc(b.x,b.y,b.r,0,Math.PI\*2); ctx.fill();  }  ctx.globalAlpha=1;  // Burbujas de la boca  for (let i=mouthBubbles.length-1;i>=0;i--){  const m=mouthBubbles[i];  m.y -= m.v\*dt;  m.life -= dt;  if(m.life<=0){ mouthBubbles.splice(i,1); continue; }  ctx.globalAlpha=Math.max(0,Math.min(1,m.life));  ctx.strokeStyle='rgba(200,220,255,.9)';  ctx.lineWidth=1.6;  ctx.beginPath(); ctx.arc(m.x,m.y,m.r,0,Math.PI\*2); ctx.stroke();  ctx.globalAlpha=1;  }  // Movimiento  const pad=72;  let desiredVx=(Math.sign(fish.vx)||1)\*fish.baseSpeed;  let desiredVy=fish.vy;  if (fish.target){  const dx=fish.target.x-fish.x, dy=fish.target.y-fish.y;  const len=Math.hypot(dx,dy)||1;  const spd=fish.followSpeed\*1.65;  desiredVx=(dx/len)\*spd;  desiredVy=(dy/len)\*spd;  } else {  fish.wobbleT += dt;  desiredVy = fish.vy + Math.sin(fish.wobbleT\*1.5)\*10\*dt;  }  // Burbujas: base + boost si turbo  fish.bubbleTimer -= dt \* (fish.target ? 2.4 : 1.0);  if (fish.bubbleTimer <= 0){  addMouthBubble(!!fish.target);  if (fish.target){  if (Math.random()<0.75) addMouthBubble(true);  if (Math.random()<0.45) addMouthBubble(true);  }  fish.bubbleTimer = rand(0.5, 1.2);  }  // Suavizado y avance  fish.vx = lerp(fish.vx, desiredVx, Math.min(1, dt\*3.0));  fish.vy = lerp(fish.vy, desiredVy, Math.min(1, dt\*3.0));  fish.x += fish.vx\*dt;  fish.y += fish.vy\*dt;  // Bordes  if (fish.x < pad){ fish.x = pad; fish.vx = Math.abs(fish.vx); }  if (fish.x > canvas.width-pad){ fish.x = canvas.width-pad; fish.vx = -Math.abs(fish.vx); }  if (fish.y < pad\*0.7){ fish.y = pad\*0.7; fish.vy = Math.abs(fish.vy); }  if (fish.y > canvas.height-pad\*0.7){ fish.y = canvas.height-pad\*0.7; fish.vy = -Math.abs(fish.vy); }  // Orientación  fish.angle = clampAngle(Math.atan2(fish.vy, Math.abs(fish.vx)));  // Dibujo  drawFish(performance.now()/1000);  requestAnimationFrame(tick);  }  requestAnimationFrame(tick);  })();  </script>  </body>  </html> |

## Adulto (franja vertical fina, aletas grandes)

|  |
| --- |
| <!doctype html>  <html lang="es">  <head>  <meta charset="utf-8" />  <meta name="viewport" content="width=device-width,initial-scale=1,maximum-scale=1,user-scalable=no"/>  <title>Adulto — franja vertical fina + aletas grandes</title>  <style>  :root { --bg:#08243a; --ui:#1d4b6b; --ui2:#2a6287; --text:#e9f6ff;}  html,body{height:100%;margin:0;background:var(--bg);}  body{font-family:system-ui,Segoe UI,Roboto,Helvetica,Arial,sans-serif;color:var(--text);overflow:hidden;}  canvas{position:fixed;inset:0;touch-action:none}  #bar{position:fixed;left:0;right:0;bottom:0;padding:10px;  display:flex;gap:10px;justify-content:center;flex-wrap:wrap;  background:linear-gradient(to top, rgba(0,0,0,.35), rgba(0,0,0,0));}  .btn{appearance:none;border:0;border-radius:12px;padding:10px 14px;  background:var(--ui);color:var(--text);font-weight:600;cursor:pointer;  box-shadow:0 6px 20px rgba(0,0,0,.25);display:inline-flex;gap:8px;align-items:center}  .btn:active{transform:translateY(1px);background:var(--ui2)}  #turbo{position:fixed;right:12px;top:12px;padding:6px 10px;border-radius:999px;  background:rgba(255,255,255,.12);backdrop-filter:blur(4px);font-weight:700;opacity:0;transition:opacity .15s}  #turbo.on{opacity:1}  </style>  </head>  <body>  <div id="turbo">TURBO 💨</div>  <canvas id="tank"></canvas>  <div id="bar">  <button class="btn" id="btnReset">↺ Reset</button>  </div>  <script>  (()=>{  const canvas=document.getElementById('tank');  const ctx=canvas.getContext('2d',{alpha:false});  function resize(){canvas.width=innerWidth;canvas.height=innerHeight;}resize();  addEventListener('resize',resize);  const turboBadge=document.getElementById('turbo');  const rand=(a,b)=>a+Math.random()\*(b-a);  const lerp=(a,b,t)=>a+(b-a)\*t;  const clampAngle=(a)=>{const m=70\*Math.PI/180;return Math.max(-m,Math.min(m,a));};  // Burbujas de fondo  const bgBubbles=Array.from({length:26},()=>({x:Math.random()\*innerWidth,y:innerHeight+Math.random()\*innerHeight,r:1+Math.random()\*3,s:12+Math.random()\*24}));  const mouthBubbles=[];  // Config del ADULTO (más grande, oscuro, franja fina, ojos más pequeños, aletas grandes)  const ADULTO={  size:60,  body:'#5a48c8', // violeta oscuro  tail:'#d073c6',  fin:'#e2a8d6',  eyeScale:0.95, // más pequeño que joven  stripeColor:'#ff9745',  stripeWidth:0.12 // fina  };  let fish={  x:canvas.width\*0.5,  y:canvas.height\*0.55,  vx:rand(90,130),  vy:rand(-22,22),  baseSpeed:125,  followSpeed:265,  wobbleT:Math.random()\*10,  angle:0,  target:null,  bubbleTimer:rand(0.25,0.9),  turbo:false  };  // Interacción: mantener pulsado = turbo + seguir dedo  let pointer=null;  canvas.addEventListener('pointerdown',e=>{pointer={x:e.clientX,y:e.clientY};fish.target=pointer;fish.turbo=true;turboBadge.classList.add('on');});  canvas.addEventListener('pointermove',e=>{if(pointer){pointer.x=e.clientX;pointer.y=e.clientY;}});  canvas.addEventListener('pointerup',()=>{pointer=null;fish.target=null;fish.turbo=false;turboBadge.classList.remove('on');});  canvas.addEventListener('pointercancel',()=>{pointer=null;fish.target=null;fish.turbo=false;turboBadge.classList.remove('on');});  document.getElementById('btnReset').onclick=()=>{  fish.x=canvas.width\*0.5; fish.y=canvas.height\*0.55;  fish.vx=rand(90,130); fish.vy=rand(-22,22);  fish.target=null; fish.turbo=false; turboBadge.classList.remove('on');  mouthBubbles.length=0;  };  function addMouthBubble(turboMul){  const dir=Math.sign(fish.vx)||1;  const mouthX=fish.x+(dir>0?ADULTO.size\*0.72:-ADULTO.size\*0.72);  mouthBubbles.push({  x:mouthX + rand(-1.0,1.0),  y:fish.y + rand(-1.0,1.0),  v:rand(22,36)\*(turboMul?1.25:1),  r:rand(1.2,2.6)\*(turboMul?1.15:1),  life:rand(0.9,1.5)\*(turboMul?1.2:1)  });  }  function drawStripeVertical(rx,ry,color,widthRatio){  // Banda vertical ajustada al óvalo (desde lomo a barriga), no se sale  const halfW=rx\*widthRatio;  const margin=2;  const rxIn=rx-margin, ryIn=ry-margin;  ctx.save(); ctx.fillStyle=color; ctx.globalAlpha=0.95; ctx.beginPath();  const steps=56;  for(let i=0;i<=steps;i++){  const t=i/steps;  const x=-halfW + t\*(2\*halfW);  const y=ryIn\*Math.sqrt(Math.max(0,1-(x\*x)/(rxIn\*rxIn)));  if(i===0) ctx.moveTo(x,-y); else ctx.lineTo(x,-y);  }  for(let i=steps;i>=0;i--){  const t=i/steps;  const x=-halfW + t\*(2\*halfW);  const y=ryIn\*Math.sqrt(Math.max(0,1-(x\*x)/(rxIn\*rxIn)));  ctx.lineTo(x,y);  }  ctx.closePath(); ctx.fill(); ctx.globalAlpha=1; ctx.restore();  }  function drawFish(time){  const s=ADULTO.size, flip=fish.vx<0?-1:1;  ctx.save(); ctx.translate(fish.x,fish.y); ctx.scale(flip,1); ctx.rotate(clampAngle(fish.angle));  // Cuerpo ovalado con contorno  const rx=s\*1.06, ry=s\*0.80;  ctx.fillStyle=ADULTO.body;  ctx.strokeStyle='rgba(0,0,0,.5)';  ctx.lineWidth=3;  ctx.beginPath(); ctx.ellipse(0,0,rx,ry,0,0,Math.PI\*2); ctx.fill(); ctx.stroke();  // Franja vertical (fina)  drawStripeVertical(rx,ry,ADULTO.stripeColor,ADULTO.stripeWidth);  // Cola (aletón más grande, aleteo suave)  const flap=Math.sin(time\*5.6 + fish.wobbleT)\*0.32;  ctx.save(); ctx.translate(-rx,0); ctx.rotate(flap);  ctx.fillStyle=ADULTO.tail;  ctx.beginPath();  ctx.moveTo(0,0);  ctx.quadraticCurveTo(-s\*0.58, -s\*0.70, s\*0.18, -s\*0.28);  ctx.quadraticCurveTo(-s\*0.58, s\*0.70, 0, 0);  ctx.fill();  ctx.restore();  // Aleta lateral GRANDE y alargada  const finFlap=Math.sin(time\*7.8 + fish.wobbleT)\*0.20;  ctx.save();  ctx.translate(-s\*0.02, s\*0.10);  ctx.rotate(finFlap);  ctx.fillStyle=ADULTO.fin;  ctx.beginPath();  ctx.moveTo(0,0);  ctx.quadraticCurveTo( s\*0.14, s\*0.95, s\*0.34, s\*0.26);  ctx.quadraticCurveTo( s\*0.10, s\*0.34, 0, 0);  ctx.fill();  ctx.restore();  // Ojo (pequeño)  const e=0.22 \* ADULTO.eyeScale, p=0.11 \* ADULTO.eyeScale;  ctx.fillStyle='#fff'; ctx.beginPath(); ctx.arc(s\*0.46, 0, s\*e, 0, Math.PI\*2); ctx.fill();  ctx.fillStyle='#111'; ctx.beginPath(); ctx.arc(s\*0.53, 0, s\*p, 0, Math.PI\*2); ctx.fill();  ctx.restore();  }  let last=performance.now();  function tick(t){  const dt=Math.min(0.033,(t-last)/1000); last=t;  // Fondo + burbujas ambientales  ctx.fillStyle='#08243a'; ctx.fillRect(0,0,canvas.width,canvas.height);  ctx.globalAlpha=0.35; ctx.fillStyle='#cfe9ff';  for(const b of bgBubbles){  b.y-=b.s\*dt;  if(b.y<-10){ b.y=canvas.height+10; b.x=Math.random()\*canvas.width; }  ctx.beginPath(); ctx.arc(b.x,b.y,b.r,0,Math.PI\*2); ctx.fill();  }  ctx.globalAlpha=1;  // Burbujas de la boca  for (let i=mouthBubbles.length-1;i>=0;i--){  const m=mouthBubbles[i];  m.y -= m.v\*dt;  m.life -= dt;  if(m.life<=0){ mouthBubbles.splice(i,1); continue; }  ctx.globalAlpha=Math.max(0,Math.min(1,m.life));  ctx.strokeStyle='rgba(200,220,255,.9)';  ctx.lineWidth=1.6;  ctx.beginPath(); ctx.arc(m.x,m.y,m.r,0,Math.PI\*2); ctx.stroke();  ctx.globalAlpha=1;  }  // Movimiento  const pad=78;  let desiredVx=(Math.sign(fish.vx)||1)\*fish.baseSpeed;  let desiredVy=fish.vy;  if (fish.target){  const dx=fish.target.x-fish.x, dy=fish.target.y-fish.y;  const len=Math.hypot(dx,dy)||1;  const spd=fish.followSpeed\*1.6;  desiredVx=(dx/len)\*spd;  desiredVy=(dy/len)\*spd;  } else {  fish.wobbleT += dt;  desiredVy = fish.vy + Math.sin(fish.wobbleT\*1.4)\*10\*dt;  }  // Burbujas: base + boost si turbo  fish.bubbleTimer -= dt \* (fish.target ? 2.2 : 1.0);  if (fish.bubbleTimer <= 0){  addMouthBubble(!!fish.target);  if (fish.target){  if (Math.random()<0.7) addMouthBubble(true);  if (Math.random()<0.4) addMouthBubble(true);  }  fish.bubbleTimer = rand(0.55, 1.3);  }  // Suavizado y avance  fish.vx = lerp(fish.vx, desiredVx, Math.min(1, dt\*3.0));  fish.vy = lerp(fish.vy, desiredVy, Math.min(1, dt\*3.0));  fish.x += fish.vx\*dt;  fish.y += fish.vy\*dt;  // Bordes  if (fish.x < pad){ fish.x = pad; fish.vx = Math.abs(fish.vx); }  if (fish.x > canvas.width-pad){ fish.x = canvas.width-pad; fish.vx = -Math.abs(fish.vx); }  if (fish.y < pad\*0.7){ fish.y = pad\*0.7; fish.vy = Math.abs(fish.vy); }  if (fish.y > canvas.height-pad\*0.7){ fish.y = canvas.height-pad\*0.7; fish.vy = -Math.abs(fish.vy); }  // Orientación  fish.angle = clampAngle(Math.atan2(fish.vy, Math.abs(fish.vx)));  // Dibujo  drawFish(performance.now()/1000);  requestAnimationFrame(tick);  }  requestAnimationFrame(tick);  })();  </script>  </body>  </html> |