

INFORME TÉCNICO COMPLETO

Free Particle FX

Estudio de Efectos de Partículas y Autómatas Celulares con 54 Presets y
Exportación con Canal Alfa



Responsable del Proyecto

Eduardo Andrés Fierro Duque

Santiago de Chile — Agosto 2026

Índice

1. Resumen Ejecutivo
2. Guía Rápida de Uso (5 pasos)
3. Stack Tecnológico
4. Arquitectura del Sistema
5. Catálogo Completo de los 54 Presets (7 Familias)
6. Fórmulas Matemáticas por Categoría de Efecto
7. Métodos Web Utilizados
8. Modelo de RAM y Rendimiento
9. Flujo de Datos Completo
10. Conclusiones

1. Resumen Ejecutivo

Free Particle FX es un estudio de efectos de partículas y autómatas celulares que funciona completamente en el navegador. Su propósito es simple y concreto: el usuario elige uno de los 54 efectos animados disponibles, ajusta unos pocos parámetros y descarga un video listo para usar en sus proyectos, con o sin fondo transparente.

Ejemplo real de uso: un animador necesita chispas y confeti para la celebración final de su escena. Abre Free Particle FX, elige el preset "Fuegos artificiales", ajusta la duración a 4 segundos, desactiva el fondo y exporta en MOV. En menos de un minuto obtiene un video con canal alfa real (ProRes 4444) que puede importar directamente en After Effects o Premiere, como haría con un material de librería profesional, pero sin instalar nada y sin costo.

Qué puede hacer concretamente la herramienta:

- Elegir entre **54 presets** organizados en 7 familias (celebración, fuego y luz, naturaleza, espacio, agua, viento y Pixel FX) con buscador integrado.
- Ajustar la simulación: cantidad de partículas, tamaño, velocidad, gravedad, viento, turbulencia, vida y opacidad, más el emisor (abajo, centro, bordes) y la mezcla (normal, aditivo, pantalla).
- Cambiar la paleta al instante: 12 paletas predefinidas o una personalizada de 5 colores, que recoloran el efecto en vivo.
- Controlar el ritmo: duración total (0.5 a 20 segundos), semilla reproducible, bucle, estelas de movimiento y pixelado global.
- Exportar en WebM (VP9 con canal alfa real), MP4 (H.264), MOV (ProRes 4444 con alfa, compatible con After Effects y Premiere) o GIF (con transparencia), a 24, 30 o 60 fps.
- Guardar y abrir proyectos (.particlefx) y autoguardado automático en el navegador.

La herramienta está disponible en producción dentro del hub de Free Animation Power (<https://freeanimationpower.org/tools/particle-fx/>) y como demo pública en GitHub Pages (<https://freeanimationpower.github.io/FREE-PARTICLE-FX/>). El código es abierto y vive en <https://github.com/freeanimationpower/FREE-PARTICLE-FX>.

2. Guía Rápida de Uso (5 pasos)

Paso 1 — Elegir el efecto: en el panel derecho se abre la familia deseada (por ejemplo "Celebracion") y se hace clic en un preset. El buscador superior del panel encuentra cualquier efecto por nombre (pruebe con "fuego" o "galaxia"). El preset aplica sus valores recomendados de cantidad, paleta y mezcla automáticamente.

Paso 2 — Ajustar la simulación: en el panel izquierdo se controlan cantidad, tamaño, velocidad, gravedad, viento, turbulencia, vida y opacidad. La barra de tiempo inferior permite reproducir, pausar (ESPACIO) y arrastrar el cursor para revisar cualquier instante.

Paso 3 — Personalizar: paleta (12 predefinidas o 5 colores propios), fondo activable con color personalizado, emisor, modo de mezcla, estelas y semilla. El botón "Aleatorio" genera una semilla nueva para variar el efecto sin tocar los demás parámetros.

Paso 4 — Guardar: el estado se autoguarda en el navegador; el botón "Guardar" descarga un archivo .particlefx (JSON) y "Abrir" lo restaura en cualquier equipo.

Paso 5 — Exportar: el botón "Exportar" abre el diálogo con formato (WebM, MP4, MOV o GIF), fps (24/30/60) y la opción "Fondo transparente". WebM, MOV y GIF conservan el canal alfa real; en MP4 la transparencia se renderiza en negro. El archivo resultante se descarga automáticamente.

3. Stack Tecnológico

Capa	Tecnología
Motor gráfico	Canvas 2D API nativa (1280×720), sin frameworks
Frontend	HTML5 + CSS3 + Vanilla JS (cero dependencias)
Backend	PHP 8 (solo el email gate del hub; todo el motor corre en el cliente)
Exportación video	MediaRecorder + canvas.captureStream(0); WebM VP9 con alfa o MP4 H.264
Exportación MOV	FFmpeg.wasm (@ffmpeg/core 0.12): ProRes 4444 con alfa (yuva444p10le), fallback qtrle (argb)
Exportación GIF	Encoder LZW propio embebido con transparencia por índice de color
Persistencia	localStorage (autoguardado) + archivos JSON .particlefx (Blob/FileReader)
Distribución	Single-file PHP (producción) + demo estática en GitHub Pages (docs/)

4. Arquitectura del Sistema

La herramienta se apoya en tres motores de render independientes que comparten un estado global, un sistema de sprites procedurales y una única función de render determinista por tiempo.

ESTADO GLOBAL (state)

Un único objeto reúne todos los parámetros editables: preset, cantidad, tamaño, velocidad, gravedad, viento, turbulencia, vida, opacidad, emisor, paleta (12 + personalizada), fondo, semilla, bucle, estelas, duración y pixelado. Cada control de la interfaz escribe en este objeto y el motor lo lee en cada frame. La firma `simSig()` une preset, semilla, paleta, viento, turbulencia, velocidad y parámetros del preset; cuando cambia, la simulación se reconstruye.

MOTOR DE PARTÍCULAS (39 presets)

`buildSim()` genera el array `SYS` de partículas al cambiar la firma. Cada partícula tiene física analítica (fórmula cerrada: caída con rozamiento exponencial, arco parabólico, rebote de muelle, órbitas); `frameP(p, tt)` devuelve posición, alfa y tamaño para un tiempo local `tt`, sin integración numérica por frame. `drawP()` dibuja con sprites pre-renderizados; las estelas de movimiento vuelven a calcular `frameP` en `tt-0.03` y `tt-0.06` con alfa menor. Cada partícula re-nace cíclicamente (`b + cyc`) para bucles continuos.

MOTOR DOOM (fuego real)

Grilla de intensidades de 160×90 (`Uint8Array`) con 37 tonos generados desde la paleta elegida (anclas oscuras → colores de paleta → blanco incandescente). Cada paso, cada celda copia la celda de abajo menos un decaimiento aleatorio y la escribe una fila arriba con desplazamiento lateral (`jitter + viento`); la fila inferior es la fuente de calor. `doomInit()` ejecuta 110 pasos de precalentado para que la llama arranque encendida. `doomDraw` pinta con `putImageData`, escala suavizado a 1280×720 y aplica una segunda pasada con `ctx.filter blur + alfa 0.55` (bloom). Intensidad 0 = alfa 0 (transparencia real).

MOTOR PIXEL FX (15 presets)

Generalización del motor DOOM: grilla `Float32Array` 160×90 + buffer auxiliar, reglas de autómata celular intercambiables, capa de intensidad, escala de alfa, precalentado y bloom. Dos familias de reglas: secuenciales (dependen del paso anterior: lava, humo, agua, arena, lluvia, matrix, tinta, ondas, estática, vida, viento) y directas (fórmula de tiempo: plasma, aurora, niebla, nebulosa).

DETERMINISMO

Todo el azar sale de `mulberry32(seed)` y `hash32(n)`. El frame `t` del preview y del export son idénticos. `render(ctx, t)` dibuja el estado exacto para cualquier instante `t`, con checkpointing de los autómatas (bloques de 16 pasos, máximo 64 entradas) para que el scrubbing hacia atrás restaure la instantánea más cercana en lugar de recalcular todo desde cero.

5. Catálogo Completo de los 54 Presets (7 Familias)

CELEBRACION (6) — partículas

Confetti (rectángulos con degradado, aleteo senoidal y sombra proyectada en el suelo); **Fuegos artificiales** (cohetes parabólicos que explotan en el vértice, 60-110 chispas con estela y destello central); **Explosion** (destello radial único con arrastre exponencial); **Fuente** (chispas hacia arriba con gravedad fuerte y rampa amarillo→naranja→rojo); **Serpentinas** (cintas poligonales de 9 segmentos ondulados, caída lenta); **Glitter** (confeti diminuto aditivo con $\sin^3$ y cruz destello en los picos).

FUEGO Y LUZ (7)

Llamas (partículas, 3 capas: cuerpo con turbulencia de 3 octavas, núcleo incandescente y chispas); **Fuego real** (autómata de calor DOOM, precalentado, bloom y paleta dinámica); **Ascuas** (ascenso con turbulencia 2-octavas y parpadeo); **Humo** (nubes suaves grises que crecen y se disipan); **Polvo magico** (espiral elíptica aditiva con cruz blanca en brillos); **Destellos** (puntos fijos con pulso $\sin^3$ y cruz de 4 rayos); **Rayos** (trayectorias precomputadas de 22 segmentos con quiebres irregulares, ramas, doble destello y resplandor de cielo).

NATURALEZA (7)

Nieve (caída lineal + sway senoidal); **Lluvia** (estelas de doble trazo en diagonal con viento); **Hojas** (sprite de hoja con vena central, giro y aleteo); **Burbujas** (círculos con degradado esférico, brillo especular y wobble); **Polen** (deriva senoidal doble + $\sin^3$ aditivo dorado); **Luciarnagas** (vagabundeo senoidal 2D con ciclo on/off y halo extra al encender); **Niebla** (sprites blancos gigantes con fundido de entrada/salida sin saltos de envoltura).

ESPACIO (7)

Campo de estrellas (estelas horizontales proporcionales a la velocidad + $\sin^3$); **Galaxia** (3 brazos espirales logarítmicos, bulge central, ~300 estrellas de fondo parpadeantes y núcleo brillante); **Bokeh** (anillos derivando lentos en modo pantalla); **Meteoros** (estela diagonal de doble trazo + cabeza blanca); **Viaje warp** (estelas radiales que crecen con el radio, efecto Star Trek, núcleo pulsante); **Aurora boreal** (5 cintas poligonales de 48 muestras con doble seno); **Cometas** (cola de 10 sprites decrecientes a lo largo del vector de movimiento).

AGUA (5) — partículas

Gotas con ondas (gota parabólica; al tocar el suelo nace un anillo elíptico con tiempo de caída precalculado); **Ondas** (anillos elípticos expansivos en la base, doble anillo); **Salpicaduras** (arcos parabólicos desde el suelo con estela y cabeza brillante); **Vapor** (columnas blancas que suben con 3 octavas de turbulencia); **Espuma** (cúmulos blancos suaves flotando desde abajo con sway).

VIENTO (7) — partículas

Rafagas (líneas curvas cuadráticas con envolvente de intensidad y doble trazo); **Torbellino** (columna cónica en espiral con altura cíclica y radio creciente); **Remolino** (vórtice en el suelo con estelas tangenciales y agujero central aditivo); **Plumas** (sprite suave alargado girando con sway amplio); **Petalos** (sprite de pétalo con aleteo y giro); **Polvo en el viento** (estelas finas horizontales de baja alfa); **Tormenta de arena** (75% granos aéreos con turbulencia + 25% granos que botan en el suelo).

PIXEL FX (15) — autómatas celulares estilo DOOM

Secuenciales: **Lava** (calor ascendente, decaimiento 0-1 viscoso); **Humo pixel** (advección ascendente con jitter lateral amplio); **Cascada pixel** (arena cayendo, se derrama lateral en el fondo); **Arena pixel** (sandpile: cae, se desliza en diagonal y se apila); **Lluvia pixel** (todo baja 2 filas/paso con decaimiento 0.96); **Matrix** (columnas con cabeza que baja y escribe intensidad 30, grilla decae 0.9); **Tinta** (difusión $(propia \cdot 0.5 + media \text{ vecinos} \cdot 0.5) \cdot 0.99$ con fuentes centrales); **Ondas pixel** (onda 2D $(\sum 4 \text{ vecinos})/2$ - anterior amortiguada 0.97, gotas cada 3 pasos); **Estatica TV** (ruido por hash con 3 filas desplazadas como banda de glitch); **Juego de la vida** (Conway B3/S23 sobre grilla toroidal, la edad de la célula mapea la paleta, re-siembra periódica de gliders); **Viento pixel** (advección de ruido por filas).

Directas (fórmula de tiempo): Plasma ($v = \sin(x \cdot 0.09 + t \cdot 1.4) + \sin(y \cdot 0.07 - t \cdot 1.1) + \sin(\text{dist} \cdot 0.09 + t \cdot 0.8)$); **Aurora pixel** (bandas senoidales en modo pantalla); **Niebla pixel** (ruido senoidal 2D lento, alfa 0.6, pantalla); **Nebulosa pixel** (3 senos multi-escala + distancia al centro, alfa 0.85, pantalla).

6. Fórmulas Matemáticas por Categoría de Efecto

6.1 Caída con rozamiento exponencial

La partícula cae con velocidad terminal: $v = (g/dr) \cdot (1 - e^{(-dr \cdot tt)})$, donde g es la gravedad y dr el coeficiente de rozamiento del aire. La posición es analítica: $y = (g/dr) \cdot tt - (g/dr^2) \cdot (1 - e^{(-dr \cdot tt)})$. No hay integración numérica: cualquier instante se evalúa con $O(1)$.

6.2 Tiempo de explosión de los cohetes

El cohete sube con velocidad inicial v_y bajo gravedad g ; alcanza el vértice en $t_e = -v_y/g$. Las chispas nacen exactamente en ese instante con la estela calculada en $tt-0.05$, lo que produce la expansión esférica característica sin simular colisiones.

6.3 Rebote de muelle (bubbles, confetti)

El rebote usa amortiguación de muelle: $v = -k \cdot y - c \cdot v$ integrada analíticamente por tramos; cada rebote reduce la altura en un factor fijo, generando la secuencia decreciente natural.

6.4 Espiral logarítmica (galaxia)

Cada estrella del brazo k se coloca con ángulo $= k \cdot (2\pi/3) + r \cdot 0.022 + \text{jitter}$ y radio aleatorio; la rotación orbital es coherente (la galaxia gira como un sólido con deriva diferencial). Las ~300 estrellas de fondo fijas parpadean con $\sin(t \cdot w + \text{fase})$.

6.5 Twinkle y pulsos (glitter, destellos, luciernagas)

El brillo pulsa con $\sin^3(t \cdot \text{speed} \cdot 3 + \text{fase})$, que aplan los mínimos y agudiza los máximos; en los picos se dibuja una cruz de 4 rayos. Las luciernagas encienden solo cuando $\sin > 0.4$, con halo extra durante el encendido.

6.6 Turbulencia multi-octava (llamas, vapor, ascuas)

El desplazamiento lateral es la suma de senos de frecuencias 1, 2 y 4 con fases derivadas de $\text{hash32}(\text{partícula}, \text{octava}, \text{semilla})$, escalada por el parámetro global de turbulencia.

6.7 Autómata de calor DOOM

$\text{nueva}[y-1][x] = \text{celda}[y][x] - \text{hash32}(\text{paso} \cdot k + \text{celda} \cdot j + \text{semilla}) \bmod 2$, con desplazamiento lateral jitter + viento; la fila inferior recibe la intensidad de la fuente. El paso se ejecuta $\text{floor}(t \cdot \text{pasos} \cdot \text{velocidad})$ veces; la paleta de 37 tonos mapea intensidad 0-35.

6.8 Onda 2D (ondas pixel)

$\text{nueva}[i][j] = (\sum 4 \text{ vecinos})/2 - \text{anterior}[i][j]$, amortiguada $\times 0.97$. Los bordes reflejan por espejo, por lo que las ondas rebotan y conservan energía. Una gota aleatoria se inyecta cada 3 pasos.

6.9 Difusión (tinta)

$\text{nueva} = (\text{propia} \cdot 0.5 + \text{media de 4 vecinos} \cdot 0.5) \cdot 0.995$, con fuente senoidal $26 + 6 \cdot \sin(\text{paso} \cdot 0.12)$ inyectada en 5 columnas con Math.max (no sobrescribe brillos existentes).

6.10 Juego de la vida

Conway B3/S23 sobre grilla toroidal de 160×90 . La edad de cada célula viva (1-35) mapea la paleta activa; cada 25 pasos se inyecta un glider (5 células) en posición derivada de $\text{hash32}(\text{paso}, \text{semilla})$ para evitar la extinción.

6.11 Barridos y clipping (estelas)

Las estelas de movimiento re-evalúan frameP en $tt-0.03$ (alfa $\times 0.3$) y $tt-0.06$ (alfa $\times 0.1$), lo que dibuja la trayectoria reciente sin almacenar estados: la memoria por frame es constante.

6.12 Captura WebM con Canal Alfa (VP9)

`stream = renderCanvas.captureStream(0);` mime = primer soportado de `['video/webm;codecs=vp9', 'video/webm']` (con alfa) o `['video/mp4;codecs=avc1', ...]` (sin alfa); `recorder = new MediaRecorder(stream, {mimeType, videoBitsPerSecond: 8000000})`. Por cada frame i ($0..duración \times fps$) se llama `render(ctx, (i/frames)·duración, mostrarFondo)` y se solicita el frame con `track.requestFrame()` (fallback: deadlines absolutos con espera activa). Con el fondo desactivado el canvas queda transparente y VP9 codifica el canal alfa real.

6.13 Encoder GIF LZW con Canal Alfa

El GIF se genera con un encoder propio (GIF89a, paleta local por frame, LZW de 8 a 12 bits): cuantización 5-6-5 bits con hasta 255 colores por frame, índice 255 reservado para transparencia (umbral de alfa < 128), emisión de CLEAR al alcanzar 4096 entradas, sub-bloques de 255 bytes y bucle NETSCAPE2.0 infinito.

6.14 Exportación MOV ProRes 4444 con FFmpeg.wasm

El MOV con transparencia (el formato nativo de After Effects y Premiere) se genera con FFmpeg compilado a WebAssembly, auto-hospedado en la carpeta `ffmpeg/` junto al `index` (el CSP del hosting bloquea CDNs) y cargado bajo demanda (~31 MB, solo la primera vez):

- Cada frame se dibuja con `render()` en el canvas offscreen y se exporta como PNG con canal alfa (`canvas.toBlob 'image/png'`), escribiéndose en el sistema de archivos virtual del WASM como `frame0001.png`, `frame0002.png`, etc.
- Se ejecuta: `ffmpeg -y -framerate fps -i frame%04d.png -c:v prores_ks -profile:v 4444 -pix_fmt yuva444p10le out.mov`
- Si el códec ProRes no está disponible en la compilación, se reintenta con QuickTime Animation: `-c:v qtrle -pix_fmt argb out.mov` (también legible por AE).

El límite por exportación es de 300 fotogramas para proteger la memoria del WASM (10 segundos a 30 fps); para clips más largos se recomienda reducir fps o duración. Los fotogramas temporales se eliminan al iniciar una nueva exportación.

7. Métodos Web Utilizados

API	Uso en la herramienta
CanvasRenderingContext2D	clearRect, fillRect, arc, save/restore, translate, rotate, scale, globalAlpha, globalCompositeOperation, filter (blur), createLinearGradient, drawImage, putImageData
mulberry32 / hash32	Generadores de azar deterministas con semilla: todo el azar del motor sale de ellos
HTMLCanvasElement.toBlob	Frames PNG con alfa para la exportación MOV
HTMLCanvasElement.captureStream	Alimenta MediaRecorder en la exportación de video
MediaRecorder	isTypeSupported negocia VP9/AVC; grabación frame a frame con requestFrame()
Blob + URL.createObjectURL	Descarga de video, MOV, GIF y proyectos
FileReader	Carga de proyectos .particlefx (JSON)
localStorage	Autoguardado continuo del estado
requestAnimationFrame	Bucle de previsualización a 60 fps
Pointer Events	Scrubbing del timeline y edición de controles
Web Worker (FFmpeg.wasm)	Codificación ProRes/qrle del MOV en el worker de @ffmpeg/ffmpeg

8. Modelo de RAM y Rendimiento

BUFFERS DE MEMORIA

- Los canvas de previsualización y exportación son de 1280×720×4 bytes = 3.69 MB cada uno. Los sprites procedurales (glow, soft, ring, grad, leaf, petal) se pre-renderizan una vez por color en texturas de 64-128 px y se cachean.
- Las grillas de los autómatas (160×90) ocupan 57 KB (DOOM, Uint8) o 115 KB (Pixel FX, Float32 con buffer auxiliar). El checkpointing usa un Map con clave sig|bloque, bloques de 16 pasos y límite de 64 entradas.
- El GIF se construye en un array de bytes con paleta local de hasta 255 colores por frame; el pico de memoria es el canvas + el array de salida.
- El MOV usa el sistema de archivos virtual del WASM: 300 PNG de 1280×720 (~1.5 MB cada uno con alfa) + el .mov resultante; por eso el límite de 300 fotogramas.

COSTO POR FRAME

- El bucle de partículas es O(n) con fórmula cerrada por partícula; con 200-400 partículas el costo es menor a 2 ms, sobrando margen para los 60 fps.
- Los autómatas secuenciales procesan 14 400 celdas por paso (40-110 pasos por segundo según el preset); el checkpointing reduce el scrub hacia atrás a 16 pasos de recálculo.
- filter blur es el atributo más costoso; solo lo usan DOOM y Pixel FX con bloom activo.

EXPORTACIÓN

- GIF: render inmediato por frame con cesión de hilo (await 0), limitado por el encoder

9. Flujo de Datos Completo

- 1. INICIALIZACIÓN:** se construye el registro de 54 presets en 7 familias (UI con buscador), se restauran el autoguardado de localStorage y el estado guardado, y se valida todo con sanitizeState() (whitelist de claves, coerción numérica con límites, preset contra PRESET_BY_ID, parámetros del preset contra sus mínimos/máximos).
- 2. RENDER CONTINUO:** requestAnimationFrame invoca render(ctx, state.time). El tiempo avanza solo con playing = true; al llegar a duration se reinicia si el bucle está activo.
- 3. EDICIÓN:** cada control escribe en el objeto state y dispara saveLS() (autoguardado). Si cambia la firma simSig() la simulación se reconstruye (buildSim, doomInit o pxlInit).
- 4. SCRUBBING:** el clic en el timeline escribe state.time y render() reconstruye el instante; los autómatas secuenciales restauran desde el checkpoint más cercano (máximo 16 pasos).
- 5. SELECCIÓN DE PRESET:** selectPreset(id) aplica los defaults del preset (cantidad, paleta, mezcla, parámetros), reconstruye la UI de parámetros dinámicos y sincroniza los controles ocultos por familia (cantidad y emisor se ocultan en Pixel FX).
- 6. GUARDAR/ABRIR:** JSON.stringify(state) → Blob → descarga .particlefx. Al abrir: FileReader → JSON.parse → sanitizeState → syncUI().
- 7. EXPORTAR:** modal con formato (WebM/MP4/MOV/GIF), fps y fondo transparente. Se renderiza frame a frame en el canvas offscreen con la MISMA función render del preview.
- 8. ACCESO:** en producción, el bloque PHP valida \$_SESSION['email'] y redirige al login del hub; en localhost (127.0.0.1) el gate se omite para desarrollo. La demo de GitHub Pages es una copia estática sin PHP.

10. Conclusiones

Free Particle FX demuestra que efectos profesionales de partículas y autómatas celulares pueden construirse sin frameworks, en un solo archivo y con calidad de exportación de estudio: 54 presets parametrizables en 7 familias, tres motores de render independientes (partículas analíticas, calor DOOM y Pixel FX), canal alfa real en WebM, GIF y MOV (ProRes 4444 listo para After Effects y Premiere), y un motor determinista que garantiza que la previsualización y el archivo exportado son idénticos.

La física analítica (fórmula cerrada en vez de integración numérica) permite evaluar cualquier instante de la simulación en $O(1)$ por partícula, lo que habilita scrubbing perfecto y exportación frame-accurate sin artefactos de tiempo real. Los autómatas celulares, por su parte, resuelven el scrubbing hacia atrás con checkpointing: la restauración repone la instantánea más cercana y recalcula como máximo 16 pasos, manteniendo el determinismo porque el azar sigue siendo $\text{hash32}(\text{paso} \cdot k + \text{celda} \cdot j + \text{semilla})$.

El patrón de doble distribución (PHP con email gate para el hub + demo estática en GitHub Pages) y la carga diferida de FFmpeg.wasm auto-hospedado (local primero, CDN de respaldo) mantienen la coherencia con el resto del ecosistema Free Animation Power: cero dependencias de servidor, cero costo para el usuario y exportación profesional desde cualquier navegador moderno.