



**EDUCACIÓN**  
SECRETARÍA DE EDUCACIÓN PÚBLICA



## **TECNOLÓGICO NACIONAL DE MÉXICO INSTITUTO TECNOLÓGICO DE TIJUANA**

**SUBDIRECCIÓN ACADÉMICA  
DEPARTAMENTO DE SISTEMAS Y COMPUTACIÓN**

**SEMESTRE:**  
Agosto - Diciembre 2025

**CARRERA:**  
Ingeniería en Sistemas Computacionales

**MATERIA:**  
Patrones de Diseño de Software

**TÍTULO ACTIVIDAD:**  
Examen unidad 4 y 5

**Alumno:**

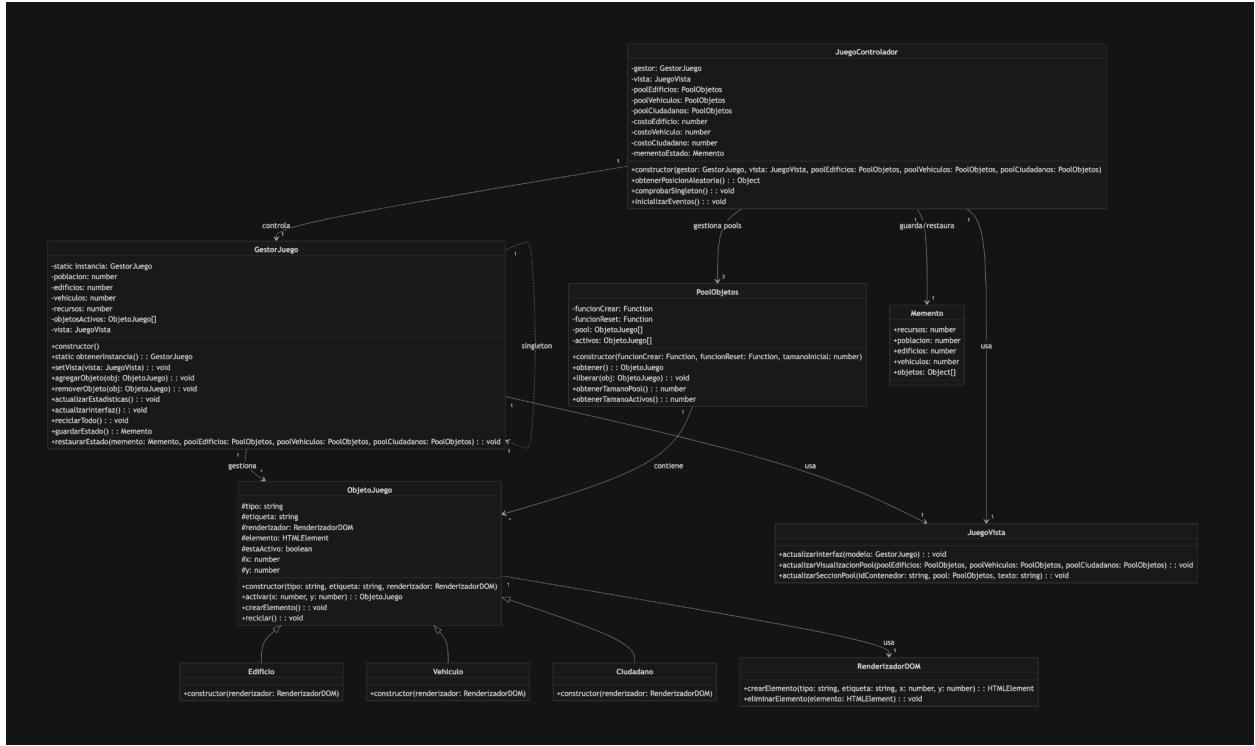
Sánchez Tolentino Gerardo Julián 21212048

**NOMBRE DEL MAESTRO(A):**  
Maribel Guerrero Luis

**Fecha**

11/12/2025

# Diagrama UML



## Código

Examen.html

```

1  <!DOCTYPE html>
2  |html lang="es">
3  |<head>
4  |    <meta charset="UTF-8">
5  |    <meta name="viewport" content="width=device-width, initial-scale=1.0">
6  |    <title>Examen de Patrones de Diseño</title>
7  |    <link rel="stylesheet" href="Examen.css">
8  |</head>
9  |<body>
10 |    <div class="contenedor-juego">
11 |        <header class="encabezado-juego">
12 |            <h1>Ciudad Peluche</h1>
13 |            <div class="estadisticas">
14 |                <div class="estadistica">
15 |                    <span class="etiqueta-estadistica">Ciudadanos:</span>
16 |                    <span id="poblacion">0</span>
17 |                </div>
18 |                <div class="estadistica">
19 |                    <span class="etiqueta-estadistica">Edificios:</span>
20 |                    <span id="edificios">0</span>
21 |                </div>
22 |                <div class="estadistica">
23 |                    <span class="etiqueta-estadistica">Vehículos:</span>
24 |                    <span id="vehiculos">0</span>
25 |                </div>
26 |                <div class="estadistica">
27 |                    <span class="etiqueta-estadistica">Recursos:</span>
28 |                    <span id="recursos">1000</span>
29 |                </div>
30 |            </div>
31 |        </header>
32 |
33 |        <main class="area-principal">
34 |            <div class="area-ciudad" id="areaCiudad">
35 |            </div>
36 |
37 |            <div class="controles">
38 |                <div class="grupo-controles">
39 |                    <button id="agregaredificio" class="boton-control">Agregar Edificio (200)</button>
40 |                    <button id="removeBuilding" class="boton-control guitar">Quitar Edificio (+100)</button>
41 |                </div>
42 |                <div class="grupo-controles">
43 |                    <button id="addVehicle" class="boton-control">Agregar Vehículo (100)</button>
44 |                    <button id="removeVehicle" class="boton-control guitar">Quitar Vehículo (+50)</button>
45 |                </div>
46 |                <div class="grupo-controles">
47 |                    <button id="agregarciudadano" class="boton-control">Agregar Ciudadano (50)</button>
48 |                    <button id="removeCitizen" class="boton-control guitar">Quitar Ciudadano (+25)</button>
49 |                </div>
50 |                <div class="grupo-controles">
51 |                    <button id="recycleAll" class="boton-control reciclar">Reciclar Todo</button>
52 |                </div>
53 |                <div class="grupo-controles">
54 |                    <button id="testSingleton" class="boton-control singleton">Probar Singleton</button>
55 |                </div>
56 |            <div class="grupo-controles">

```

```
57 |         <button id="guardarEstado" class="boton-control memento">Guardar Estado</button>
58 |         <button id="restaurarEstado" class="boton-control memento">Restaurar Estado</button>
59 |
60 |     </div>
61 | </main>
62 |
63 | <div class="panel-pool-objetos" id="objectPoolPanel">
64 |     <h3>Pool de Objetos</h3>
65 |     <div class="seccion-pool">
66 |         <h4>Edificios Disponibles</h4>
67 |         <div class="items-pool" id="edificioPool">
68 |             </div>
69 |         </div>
70 |         <div class="seccion-pool">
71 |             <h4>Vehículos Disponibles</h4>
72 |             <div class="items-pool" id="vehiculoPool">
73 |                 </div>
74 |             </div>
75 |             <div class="seccion-pool">
76 |                 <h4>Ciudadanos Disponibles</h4>
77 |                 <div class="items-pool" id="ciudadanoPool">
78 |                     </div>
79 |                 </div>
80 |             </div>
81 |         </div>
82 |
83 |     <script>
84 |         class GestorJuego
85 |     {
86 |         constructor()
87 |     {
88 |         if (GestorJuego.instancia)
89 |         {
90 |             return GestorJuego.instancia;
91 |         }
92 |         this.poblacion = 0;
93 |         this.edificios = 0;
94 |         this.vehiculos = 0;
95 |         this.recursos = 1000;
96 |         this.objetosActivos = [];
97 |         this.vista = null;
98 |
99 |         GestorJuego.instancia = this;
100 |         this.actualizarInterfaz();
101 |     }
102 |
103 |     static obtenerInstancia()
104 |     {
105 |         if (!GestorJuego.instancia)
106 |         {
107 |             GestorJuego.instancia = new GestorJuego();
```

```
107     GestorJuego.instancia = new GestorJuego();
108 }
109     return GestorJuego.instancia;
110 }
111
112     setVista(vista)
113 {
114     this.vista = vista;
115     this.actualizarInterfaz();
116 }
117
118     agregarObjeto(obj) //Metodo para agregar un objeto al gestor de juego
119 {
120     this.objetosActivos.push(obj);
121     this.actualizarEstadisticas();
122     this.actualizarInterfaz();
123 }
124
125     removerObjeto(obj)
126 {
127     const indice = this.objetosActivos.indexOf(obj);
128     if (indice > -1)
129     {
130         this.objetosActivos.splice(indice, 1);
131         this.actualizarEstadisticas();
132         this.actualizarInterfaz();
133     }
134 }
135
136     actualizarEstadisticas()
137 {
138     this.poblacion = this.objetosActivos.filter(obj => obj.tipo === 'ciudadano').length;
139     this.edificios = this.objetosActivos.filter(obj => obj.tipo === 'edificio').length;
140     this.vehiculos = this.objetosActivos.filter(obj => obj.tipo === 'vehiculo').length;
141 }
142
143     actualizarInterfaz()
144 {
145     if (this.vista)
146     {
147         this.vista.actualizarInterfaz(this);
148     }
149     else
150     {
151         document.getElementById('poblacion').textContent = this.poblacion;
152         document.getElementById('edificios').textContent = this.edificios;
153         document.getElementById('vehiculos').textContent = this.vehiculos;
154         document.getElementById('recursos').textContent = this.recursos;
155     }
156 }
```

```
157     reciclarTodo()
158     {
159         this.objetosActivos.forEach(obj =>
160         {
161             obj.reciclar();
162         });
163         this.objetosActivos = [];
164         this.actualizarEstadisticas();
165         this.actualizarInterfaz();
166     }
167
168
169     guardarEstado()
170     {
171         const objetosInfo = this.objetosActivos.map(obj => ({
172             tipo: obj.tipo,
173             x: obj.x,
174             y: obj.y
175         }));
176         return {
177             recursos: this.recursos,
178             poblacion: this.poblacion,
179             edificios: this.edificios,
180             vehiculos: this.vehiculos,
181             objetos: objetosInfo
182         };
183     }
184
185     restaurarEstado(memento, poolEdificios, poolVehiculos, poolCiudadanos)
186     {
187         this.objetosActivos.forEach(obj => {
188             obj.reciclar();
189             if (obj.tipo === 'edificio') {
190                 poolEdificios.liberar(obj);
191             } else if (obj.tipo === 'vehiculo') {
192                 poolVehiculos.liberar(obj);
193             } else if (obj.tipo === 'ciudadano') {
194                 poolCiudadanos.liberar(obj);
195             }
196         });
197         this.objetosActivos = [];
198
199         this.recursos = memento.recursos;
200         this.poblacion = memento.poblacion;
201         this.edificios = memento.edificios;
202         this.vehiculos = memento.vehiculos;
203
204         if (memento.objetos) {
205             memento.objetos.forEach(objInfo => {
206                 let obj = null;
207                 if (objInfo.tipo === 'edificio') {
208                     obj = poolEdificios.obtener();
```

```
208         obj = poolEdificios.obtener();
209     } else if (objInfo.tipo === 'vehiculo') {
210         obj = poolVehiculos.obtener();
211     } else if (objInfo.tipo === 'ciudadano') {
212         obj = poolCiudadanos.obtener();
213     }
214
215     if (obj) {
216         obj.activar(objInfo.x, objInfo.y);
217         this.agregarObjeto(obj);
218     }
219 }
220 }
221
222     this.actualizarInterfaz();
223 }
224 }
225
226 class PoolObjetos
227 {
228     constructor(funcionCrear, funcionReset, tamanoInicial = 5)
229     {
230         this.funcionCrear = funcionCrear;
231         this.funcionReset = funcionReset;
232         this.pool = [];
233         this.activos = [];
234
235         for (let i = 0; i < tamanoInicial; i++)
236         {
237             this.pool.push(this.funcionCrear());
238         }
239     }
240
241     obtener()
242     {
243         if (this.pool.length === 0)
244         {
245             return null;
246         }
247
248         const obj = this.pool.pop();
249         this.activos.push(obj);
250         return obj;
251     }
252
253     liberar(obj)
254     {
255         const indice = this.activos.indexOf(obj);
256         if (indice > -1)
257         {
258             this.activos.splice(indice, 1);
259             this.pool.push(obj);
260         }
261     }
262 }
```

```
258         this.activos.splice(indice, 1); //Quita de activos el objeto
259         this.funcionReset(obj); //Llama a la función de reset del objeto
260         this.pool.push(obj); //Regresa el objeto al pool
261     }
262 }
263
264     obtenerTamanoPool()
265     {
266         return this.pool.length;
267     }
268
269     obtenerTamanoActivos()
270     {
271         return this.activos.length;
272     }
273 }
274
275 class RenderizadorDOM
276 {
277     crearElemento(tipo, etiqueta, x, y)
278     {
279         const elemento = document.createElement('div');
280         elemento.className = tipo;
281         elemento.style.left = x + 'px';
282         elemento.style.top = y + 'px';
283         elemento.innerHTML = etiqueta;
284         document.getElementById('areaCiudad').appendChild(elemento);
285         return elemento;
286     }
287
288     eliminarElemento(elemento)
289     {
290         if (elemento)
291         {
292             elemento.remove();
293         }
294     }
295 }
296
297 class ObjetoJuego
298 {
299     constructor(tipo, etiqueta, renderizador)
300     {
301         this.tipo = tipo;
302         this.etiqueta = etiqueta;
303         this.renderizador = renderizador;
304         this.elemento = null;
305         this.estaActivo = false;
306         this.x = 0;
307         this.y = 0;
308     }
```

```
309
310     activar(x, y)
311     {
312         this.estaActivo = true;
313         this.x = x;
314         this.y = y;
315         this.crearElemento();
316         return this;
317     }
318
319     crearElemento()
320     {
321         this.elemento = this.renderizador.crearElemento(this.tipo, this.etiqueta, this.x, this.y);
322     }
323
324     reciclar()
325     {
326         this.renderizador.eliminarElemento(this.elemento);
327         this.estaActivo = false;
328         this.elemento = null;
329     }
330 }
331
332 class Edificio extends ObjetoJuego
333 {
334     constructor(renderizador)
335     {
336         super('edificio', 'EDIFICIO', renderizador);
337     }
338 }
339
340 class Vehiculo extends ObjetoJuego
341 {
342     constructor(renderizador)
343     {
344         super('vehiculo', 'VEHÍCULO', renderizador);
345     }
346 }
347
348 class Ciudadano extends ObjetoJuego
349 {
350     constructor(renderizador)
351     {
352         super('ciudadano', 'CIUDADANO', renderizador);
353     }
354 }
355
356 // ===== Vista (View) =====
357 class JuegoVista
358 {
359     actualizarInterfaz(modelo)
```

```

360     {
361         document.getElementById('poblacion').textContent = modelo.poblacion;
362         document.getElementById('edificios').textContent = modelo.edificios;
363         document.getElementById('vehiculos').textContent = modelo.vehiculos;
364         document.getElementById('recursos').textContent = modelo.recursos;
365     }
366
367     actualizarVisualizacionPool(poolEdificios, poolVehiculos, poolCiudadanos)
368     {
369         this.actualizarSeccionPool('edificioPool', poolEdificios, 'EDIFICIO');
370         this.actualizarSeccionPool('vehiculoPool', poolVehiculos, 'VEHÍCULO');
371         this.actualizarSeccionPool('ciudadanoPool', poolCiudadanos, 'CIUDADANO');
372     }
373
374     actualizarSeccionPool(idContenedor, pool, texto)
375     {
376         const contenedor = document.getElementById(idContenedor);
377         contenedor.innerHTML = '';
378
379         for (let i = 0; i < pool.obtenerTamañoPool(); i++)
380         {
381             const itemPool = document.createElement('div');
382             itemPool.className = 'item-pool';
383             itemPool.innerHTML = texto;
384             contenedor.appendChild(itemPool);
385         }
386     }
387 }
388 class JuegoControlador
389 {
390     constructor(gestor, vista, poolEdificios, poolVehiculos, poolCiudadanos)
391     {
392         this.gestor = gestor;
393         this.vista = vista;
394         this.poolEdificios = poolEdificios;
395         this.poolVehiculos = poolVehiculos;
396         this.poolCiudadanos = poolCiudadanos;
397         this.costoEdificio = 200;
398         this.costoVehiculo = 100;
399         this.costoCiudadano = 50;
400         this.mementoEstado = null;
401         this.inicializarEventos();
402         this.vista.actualizarVisualizacionPool(this.poolEdificios, this.poolVehiculos, this.poolCiudadanos);
403     }
404
405     obtenerPosicionAleatoria()
406     {
407         const areaCiudad = document.getElementById('areaCiudad');
408         const rectangulo = areaCiudad.getBoundingClientRect();
409         return {
410             x: Math.random() * (rectangulo.width - 50),

```

```
411         y: Math.random() * (rectangulo.height - 50)
412     );
413 }
414
415 comprobarSingleton()
416 {
417     const instanciaUno = GestorJuego.obtenerInstancia();
418     const instanciaDos = GestorJuego.obtenerInstancia();
419     const esSingleton = (instanciaUno === instanciaDos);
420
421     const mensaje = `COMPROBACIÓN DE SINGLETON:\n\n` +
422         `¿Son la misma instancia? ${esSingleton} ? 'SÍ' : 'NO'\n\n` +
423         `Singleton OK: ${esSingleton}\n\n`;
424
425     alert(mensaje);
426 }
427
428 inicializarEventos()
429 {
430     document.getElementById('agregaredificio').addEventListener('click', () =>
431     {
432         if (this.gestor.recursos < this.costoEdificio)
433         {
434             return;
435         }
436
437         const posicion = this.obtenerPosicionAleatoria();
438         const edificio = this.poolEdificios.obtener();
439
440         if (edificio === null)
441         {
442             return;
443         }
444
445         this.gestor.recursos -= this.costoEdificio;
446         edificio.activar(posicion.x, posicion.y);
447         this.gestor.agregarObjeto(edificio);
448         this.gestor.actualizarInterfaz();
449         this.vista.actualizarVisualizacionPool(this.poolEdificios, this.poolVehiculos, this.poolCiudadanos);
450     });
451
452     document.getElementById('addVehicle').addEventListener('click', () =>
453     {
454         if (this.gestor.recursos < this.costoVehiculo)
455         {
456             return;
457         }
458
459         const posicion = this.obtenerPosicionAleatoria();
460         const vehiculo = this.poolVehiculos.obtener();
```

```
461
462     if (vehiculo === null)
463     {
464         return;
465     }
466
467     this.gestor.recursos -= this.costoVehiculo;
468     vehiculo.activar(posicion.x, posicion.y);
469     this.gestor.agregarObjeto(vehiculo);
470     this.gestor.actualizarInterfaz();
471     this.vista.actualizarVisualizacionPool(this.poolEdificios, this.poolVehiculos, this.poolCiudadanos);
472 );
473
474 document.getElementById('agregarciudadano').addEventListener('click', () =>
475 {
476     if (this.gestor.recursos < this.costoCiudadano)
477     {
478         return;
479     }
480
481     const posicion = this.obtenerPosicionAleatoria();
482     const ciudadano = this.poolCiudadanos.obtener();
483
484     if (ciudadano === null)
485     {
486         return;
487     }
488
489     this.gestor.recursos -= this.costoCiudadano;
490     ciudadano.activar(posicion.x, posicion.y);
491     this.gestor.agregarObjeto(ciudadano);
492     this.gestor.actualizarInterfaz();
493     this.vista.actualizarVisualizacionPool(this.poolEdificios, this.poolVehiculos, this.poolCiudadanos);
494 );
495
496 document.getElementById('removeBuilding').addEventListener('click', () => {
497     const edificiosActivos = this.gestor.objetosActivos.filter(obj => obj.tipo === 'edificio');
498
499     if (edificiosActivos.length === 0)
500     {
501         return;
502     }
503
504     const edificioAQuitar = edificiosActivos[edificiosActivos.length - 1];
505     this.gestor.recursos += Math.floor(this.costoEdificio * 0.5);
506     this.poolEdificios.liberar(edificioAQuitar);
507     this.gestor.removerObjeto(edificioAQuitar);
508     this.gestor.actualizarInterfaz();
509     this.vista.actualizarVisualizacionPool(this.poolEdificios, this.poolVehiculos, this.poolCiudadanos);
510 });
511
```

```
511
512
513
514
515
516
517
518
519
520
521
522
523
524
525
526
527
528
529
530
531
532
533
534
535
536
537
538
539
540
541
542
543
544
545
546
547
548
549
550
551
552
553
554
555
556
557
558
559
```

```
document.getElementById('removeVehicle').addEventListener('click', () =>
{
    const vehiculosActivos = this.gestor.objetosActivos.filter(obj => obj.tipo === 'vehiculo');

    if (vehiculosActivos.length === 0)
    {
        return;
    }

    const vehiculoAQuitar = vehiculosActivos[vehiculosActivos.length - 1];
    this.gestor.recursos += Math.floor(this.costoVehiculo * 0.5);
    this.poolVehiculos.liberar(vehiculoAQuitar);
    this.gestor.removerObjeto(vehiculoAQuitar);
    this.gestor.actualizarInterfaz();
    this.vista.actualizarVisualizacionPool(this.poolEdificios, this.poolVehiculos, this.poolCiudadanos);
});

document.getElementById('removeCitizen').addEventListener('click', () =>
{
    const ciudadanosActivos = this.gestor.objetosActivos.filter(obj => obj.tipo === 'ciudadano');

    if (ciudadanosActivos.length === 0)
    {
        return;
    }

    const ciudadanoAQuitar = ciudadanosActivos[ciudadanosActivos.length - 1];
    this.gestor.recursos += Math.floor(this.costoCiudadano * 0.5);
    this.poolCiudadanos.liberar(ciudadanoAQuitar);
    this.gestor.removerObjeto(ciudadanoAQuitar);
    this.gestor.actualizarInterfaz();
    this.vista.actualizarVisualizacionPool(this.poolEdificios, this.poolVehiculos, this.poolCiudadanos);
});

document.getElementById('recycleAll').addEventListener('click', () =>
{
    let recursosADevolver = 0;
    this.gestor.objetosActivos.forEach(obj =>
    {
        if (obj.tipo === 'edificio')
        {
            recursosADevolver += Math.floor(this.costoEdificio * 0.5);
        } else if (obj.tipo === 'vehiculo')
        {
            recursosADevolver += Math.floor(this.costoVehiculo * 0.5);
        } else if (obj.tipo === 'ciudadano')
        {
            recursosADevolver += Math.floor(this.costoCiudadano * 0.5);
        }
    });
});
```

```
561
562
563     this.gestor.objetosActivos.forEach(obj => {
564         if (obj.tipo === 'edificio')
565         {
566             this.poolEdificios.liberar(obj);
567         } else if (obj.tipo === 'vehiculo')
568         {
569             this.poolVehiculos.liberar(obj);
570         } else if (obj.tipo === 'ciudadano')
571         {
572             this.poolCiudadanos.liberar(obj);
573         }
574     });
575
576     this.gestor.recursos += recursosADevolver;
577     this.gestor.reciclarTodo();
578     this.gestor.actualizarInterfaz();
579     this.vista.actualizarVisualizacionPool(this.poolEdificios, this.poolVehiculos, this.poolCiudadanos);
580 });
581
582 document.getElementById('testSingleton').addEventListener('click', () =>
583 {
584     this.comprobarSingleton();
585 });
586
587 document.getElementById('guardarEstado').addEventListener('click', () =>
588 {
589     this.mementoEstado = this.gestor.guardarEstado();
590     alert('Estado guardado correctamente');
591 });
592
593 document.getElementById('restaurarEstado').addEventListener('click', () =>
594 {
595     if (this.mementoEstado)
596     {
597         this.gestor.restaurarEstado(this.mementoEstado, this.poolEdificios, this.poolVehiculos, this.poolCiudadanos);
598         this.vista.actualizarVisualizacionPool(this.poolEdificios, this.poolVehiculos, this.poolCiudadanos);
599         alert('Estado restaurado correctamente');
600     }
601     else
602     {
603         alert('No hay estado guardado');
604     }
605 });
606
607 });
608
609 const gestorJuego = GestorJuego.obtenerInstancia();
610 const renderizadorDOM = new RenderizadorDOM();
```

```
599         alert('Estado restaurado correctamente');
600     }
601     else
602     {
603         alert('No hay estado guardado');
604     }
605 });
606 }
607 }
608
609 const gestorJuego = GestorJuego.obtenerInstancia();
610 const renderizadorDOM = new RenderizadorDOM();
611 const vista = new JuegoVista();
612 gestorJuego.setVista(vista);
613
614 const poolEdificios = new PoolObjetos(
615     () => new Edificio(renderizadorDOM),
616     (obj) => obj.reciclar(),
617     5
618 );
619
620 const poolVehiculos = new PoolObjetos(
621     () => new Vehiculo(renderizadorDOM),
622     (obj) => obj.reciclar(),
623     5
624 );
625
626 const poolCiudadanos = new PoolObjetos(
627     () => new Ciudadano(renderizadorDOM),
628     (obj) => obj.reciclar(),
629     5
630 );
631
632 new JuegoControlador(gestorJuego, vista, poolEdificios, poolVehiculos, poolCiudadanos);
633 vista.actualizarVisualizacionPool(poolEdificios, poolVehiculos, poolCiudadanos);
634 </script>
635 </body>
636 </html>
```

## [Examen.js](#)

```
1  class GestorJuego
2  {
3      constructor()
4      {
5          if (GestorJuego.instancia)
6          {
7              return GestorJuego.instancia;
8          }
9          this.poblacion = 0;
10         this.edificios = 0;
11         this.vehiculos = 0;
12         this.recursos = 1000;
13         this.objetosActivos = [];
14         this.vista = null;
15
16         GestorJuego.instancia = this;
17         this.actualizarInterfaz();
18     }
19
20     static obtenerInstancia()
21     {
22         if (!GestorJuego.instancia)
23         {
24             GestorJuego.instancia = new GestorJuego();
25         }
26         return GestorJuego.instancia;
27     }
28
29     setVista(vista)
30     {
31         this.vista = vista;
32         this.actualizarInterfaz();
33     }
34
35     agregarObjeto(obj)
36     {
37         this.objetosActivos.push(obj);
38         this.actualizarEstadisticas();
39         this.actualizarInterfaz();
40     }
41
42     removerObjeto(obj)
43     {
44         const indice = this.objetosActivos.indexOf(obj);
45         if (indice > -1)
46         {
47             this.objetosActivos.splice(indice, 1);
48             this.actualizarEstadisticas();
49             this.actualizarInterfaz();
50         }
51     }
52
53     actualizarEstadisticas()
54     {
55         this.poblacion = this.objetosActivos.filter(obj => obj.tipo === 'ciudadano').length;
56         this.edificios = this.objetosActivos.filter(obj => obj.tipo === 'edificio').length;
```

```
57     |     this.vehiculos = this.objetosActivos.filter(obj => obj.tipo === 'vehiculo').length;
58   }
59
60   actualizarInterfaz()
61   {
62     if (this.vista)
63     {
64       this.vista.actualizarInterfaz(this);
65     }
66     else
67     {
68       document.getElementById('poblacion').textContent = this.poblacion;
69       document.getElementById('edificios').textContent = this.edificios;
70       document.getElementById('vehiculos').textContent = this.vehiculos;
71       document.getElementById('recursos').textContent = this.recursos;
72     }
73   }
74
75   reciclarTodo()
76   {
77     this.objetosActivos.forEach(obj =>
78     {
79       obj.reciclar();
80     });
81     this.objetosActivos = [];
82     this.actualizarEstadisticas();
83     this.actualizarInterfaz();
84   }
85
86   guardarEstado()
87   {
88     const objetosInfo = this.objetosActivos.map(obj => ({
89       tipo: obj.tipo,
90       x: obj.x,
91       y: obj.y
92     }));
93     return {
94       recursos: this.recursos,
95       poblacion: this.poblacion,
96       edificios: this.edificios,
97       vehiculos: this.vehiculos,
98       objetos: objetosInfo
99     };
100   }
101
102   restaurarEstado(memento, poolEdificios, poolVehiculos, poolCiudadanos)
103   {
104     this.objetosActivos.forEach(obj => {
105       obj.reciclar();
106       if (obj.tipo === 'edificio') {
107         poolEdificios.liberar(obj);
108       } else if (obj.tipo === 'vehiculo') {
109         poolVehiculos.liberar(obj);
110       } else if (obj.tipo === 'ciudadano') {
```

```
111         poolCiudadanos.liberar(obj);
112     }
113 });
114 this.objetosActivos = [];
115
116 this.recursos = memento.recursos;
117 this.poblacion = memento.poblacion;
118 this.edificios = memento.edificios;
119 this.vehiculos = memento.vehiculos;
120
121 if (memento.objetos) {
122     memento.objetos.forEach(objInfo => {
123         let obj = null;
124         if (objInfo.tipo === 'edificio') {
125             obj = poolEdificios.obtener();
126         } else if (objInfo.tipo === 'vehiculo') {
127             obj = poolVehiculos.obtener();
128         } else if (objInfo.tipo === 'ciudadano') {
129             obj = poolCiudadanos.obtener();
130         }
131
132         if (obj) {
133             obj.activar(objInfo.x, objInfo.y);
134             this.agregarObjeto(obj);
135         }
136     });
137 }
138
139 this.actualizarInterfaz();
140 }
141 }
142
143 class PoolObjetos
144 {
145     constructor(funcionCrear, funcionReset, tamanoInicial = 5)
146     {
147         this.funcionCrear = funcionCrear;
148         this.funcionReset = funcionReset;
149         this.pool = [];
150         this.activos = [];
151
152         for (let i = 0; i < tamanoInicial; i++)
153         {
154             this.pool.push(this.funcionCrear());
155         }
156     }
157
158     obtener()
159     {
160         if (this.pool.length === 0)
161         {
162             return null;
163         }
164     }
165 }
```

```
164     const obj = this.pool.pop();
165     this.activos.push(obj);
166     return obj;
167   }
168
169   liberar(obj)
170   {
171     const indice = this.activos.indexOf(obj);
172     if (indice > -1)
173     {
174       this.activos.splice(indice, 1);
175       this.funcionReset(obj);
176       this.pool.push(obj);
177     }
178   }
179
180
181   obtenerTamanoPool()
182   {
183     return this.pool.length;
184   }
185
186   obtenerTamanoActivos()
187   {
188     return this.activos.length;
189   }
190 }
191
192 class RenderizadorDOM
193 {
194   crearElemento(tipo, etiqueta, x, y)
195   {
196     const elemento = document.createElement('div');
197     elemento.className = tipo;
198     elemento.style.left = x + 'px';
199     elemento.style.top = y + 'px';
200     elemento.innerHTML = etiqueta;
201     document.getElementById('areaCiudad').appendChild(elemento);
202     return elemento;
203   }
204
205   eliminarElemento(elemento)
206   {
207     if (elemento)
208     {
209       elemento.remove();
210     }
211   }
212 }
213
214 class ObjetoJuego
215 {
216   constructor(tipo, etiqueta, renderizador)
217   {
```

```
218     this.tipo = tipo;
219     this.etiqueta = etiqueta;
220     this.renderizador = renderizador;
221     this.elemento = null;
222     this.estaActivo = false;
223     this.x = 0;
224     this.y = 0;
225   }
226
227   activar(x, y)
228   {
229     this.estaActivo = true;
230     this.x = x;
231     this.y = y;
232     this.crearElemento();
233     return this;
234   }
235
236   crearElemento()
237   {
238     this.elemento = this.renderizador.crearElemento(this.tipo, this.etiqueta, this.x, this.y);
239   }
240
241   reciclar()
242   {
243     this.renderizador.eliminarElemento(this.elemento);
244     this.estaActivo = false;
245     this.elemento = null;
246   }
247 }
248
249 class Edificio extends ObjetoJuego
250 {
251   constructor(renderizador)
252   {
253     super('edificio', 'EDIFICIO', renderizador);
254   }
255 }
256
257 class Vehiculo extends ObjetoJuego
258 {
259   constructor(renderizador)
260   {
261     super('vehiculo', 'VEHÍCULO', renderizador);
262   }
263 }
264
265 class Ciudadano extends ObjetoJuego
266 {
267   constructor(renderizador)
268   {
269     super('ciudadano', 'CIUDADANO', renderizador);
270   }
271 }
```

```

274  class JuegoVista
275  {
276      actualizarInterfaz(modelo)
277      {
278          document.getElementById('poblacion').textContent = modelo.poblacion;
279          document.getElementById('edificios').textContent = modelo.edificios;
280          document.getElementById('vehiculos').textContent = modelo.vehiculos;
281          document.getElementById('recursos').textContent = modelo.recursos;
282      }
283
284      actualizarVisualizacionPool(poolEdificios, poolVehiculos, poolCiudadanos)
285      {
286          this.actualizarSeccionPool('edificioPool', poolEdificios, 'EDIFICIO');
287          this.actualizarSeccionPool('vehiculoPool', poolVehiculos, 'VEHICULO');
288          this.actualizarSeccionPool('ciudadanoPool', poolCiudadanos, 'CIUDADANO');
289      }
290
291      actualizarSeccionPool(idContenedor, pool, texto)
292      {
293          const contenedor = document.getElementById(idContenedor);
294          contenedor.innerHTML = '';
295
296          for (let i = 0; i < pool.obtenerTamanoPool(); i++)
297          {
298              const itemPool = document.createElement('div');
299              itemPool.className = 'item-pool';
300              itemPool.innerHTML = texto;
301              contenedor.appendChild(itemPool);
302          }
303      }
304  }
305
306 // ===== Controller =====
307 class JuegoControlador
308 {
309     constructor(gestor, vista, poolEdificios, poolVehiculos, poolCiudadanos)
310     {
311         this.gestor = gestor;
312         this.vista = vista;
313         this.poolEdificios = poolEdificios;
314         this.poolVehiculos = poolVehiculos;
315         this.poolCiudadanos = poolCiudadanos;
316         this.costoEdificio = 200;
317         this.costoVehiculo = 100;
318         this.costoCiudadano = 50;
319         this.mementoEstado = null;
320         this.inicializarEventos();
321         this.vista.actualizarVisualizacionPool(this.poolEdificios, this.poolVehiculos, this.poolCiudadanos);
322     }
323
324     obtenerPosicionAleatoria()
325     {
326         const areaCiudad = document.getElementById('areaCiudad');

```

```
327     const rectangulo = areaCiudad.getBoundingClientRect();
328     return {
329       x: Math.random() * (rectangulo.width - 50),
330       y: Math.random() * (rectangulo.height - 50)
331     };
332   }
333
334   comprobarSingleton()
335   {
336     const instanciaUno = GestorJuego.obtenerInstancia();
337     const instanciaDos = GestorJuego.obtenerInstancia();
338     const esSingleton = (instanciaUno === instanciaDos);
339
340     const mensaje = `COMPROBACIÓN DE SINGLETON:\n\n` +
341       `¿Son la misma instancia? ${esSingleton} ? 'SÍ' : 'NO'\n\n` +
342       `Singleton OK: ${esSingleton}\n\n`;
343
344     alert(mensaje);
345   }
346
347   inicializarEventos()
348   {
349     document.getElementById('agregaredificio').addEventListener('click', () =>
350     {
351       if (this.gestor.recursos < this.costoEdificio)
352       {
353         return;
354       }
355
356       const posicion = this.obtenerPosicionAleatoria();
357       const edificio = this.poolEdificios.obtener();
358
359       if (edificio === null)
360       {
361         return;
362       }
363
364       this.gestor.recursos -= this.costoEdificio;
365       edificio.activar(posicion.x, posicion.y);
366       this.gestor.agregarObjeto(edificio);
367       this.gestor.actualizarInterfaz();
368       this.vista.actualizarVisualizacionPool(this.poolEdificios, this.poolVehiculos, this.poolCiudadanos);
369     });
370
371     document.getElementById('addVehicle').addEventListener('click', () =>
372     {
373       if (this.gestor.recursos < this.costoVehiculo)
374       {
375         return;
376       }
377
378       const posicion = this.obtenerPosicionAleatoria();
379       const vehiculo = this.poolVehiculos.obtener();
380     });
381   }
382 }
```

```
381     if (vehiculo === null)
382     {
383         return;
384     }
385
386     this.gestor.recursos -= this.costoVehiculo;
387     vehiculo.activar(posicion.x, posicion.y);
388     this.gestor.agregarObjeto(vehiculo);
389     this.gestor.actualizarInterfaz();
390     this.vista.actualizarVisualizacionPool(this.poolEdificios, this.poolVehiculos, this.poolCiudadanos);
391 });
392
393 document.getElementById('agregarciudadano').addEventListener('click', () =>
394 {
395     if (this.gestor.recursos < this.costoCiudadano)
396     {
397         return;
398     }
399
400     const posicion = this.obtenerPosicionAleatoria();
401     const ciudadano = this.poolCiudadanos.obtener();
402
403     if (ciudadano === null)
404     {
405         return;
406     }
407
408     this.gestor.recursos -= this.costoCiudadano;
409     ciudadano.activar(posicion.x, posicion.y);
410     this.gestor.agregarObjeto(ciudadano);
411     this.gestor.actualizarInterfaz();
412     this.vista.actualizarVisualizacionPool(this.poolEdificios, this.poolVehiculos, this.poolCiudadanos);
413 });
414
415 document.getElementById('removeBuilding').addEventListener('click', () => {
416     const edificiosActivos = this.gestor.objetosActivos.filter(obj => obj.tipo === 'edificio');
417
418     if (edificiosActivos.length === 0)
419     {
420         return;
421     }
422
423     const edificioAQuitar = edificiosActivos[edificiosActivos.length - 1];
424     this.gestor.recursos += Math.floor(this.costoEdificio * 0.5);
425     this.poolEdificios.liberar(edificioAQuitar);
426     this.gestor.removerObjeto(edificioAQuitar);
427     this.gestor.actualizarInterfaz();
428     this.vista.actualizarVisualizacionPool(this.poolEdificios, this.poolVehiculos, this.poolCiudadanos);
429 });
430
431 document.getElementById('removeVehicle').addEventListener('click', () =>
```

```
431 document.getElementById('removeVehicle').addEventListener('click', () =>
432 {
433     const vehiculosActivos = this.gestor.objetosActivos.filter(obj => obj.tipo === 'vehiculo');
434
435     if (vehiculosActivos.length === 0)
436     {
437         return;
438     }
439
440     const vehiculoAQuitar = vehiculosActivos[vehiculosActivos.length - 1];
441     this.gestor.recursos += Math.floor(this.costoVehiculo * 0.5);
442     this.poolVehiculos.liberar(vehiculoAQuitar);
443     this.gestor.removerObjeto(vehiculoAQuitar);
444     this.gestor.actualizarInterfaz();
445     this.vista.actualizarVisualizacionPool(this.poolEdificios, this.poolVehiculos, this.poolCiudadanos);
446 });
447
448 document.getElementById('removeCitizen').addEventListener('click', () =>
449 {
450     const ciudadanosActivos = this.gestor.objetosActivos.filter(obj => obj.tipo === 'ciudadano');
451
452     if (ciudadanosActivos.length === 0)
453     {
454         return;
455     }
456
457     const ciudadanoAQuitar = ciudadanosActivos[ciudadanosActivos.length - 1];
458     this.gestor.recursos += Math.floor(this.costoCiudadano * 0.5);
459     this.poolCiudadanos.liberar(ciudadanoAQuitar);
460     this.gestor.removerObjeto(ciudadanoAQuitar);
461     this.gestor.actualizarInterfaz();
462     this.vista.actualizarVisualizacionPool(this.poolEdificios, this.poolVehiculos, this.poolCiudadanos);
463 });
464
465 document.getElementById('recycleAll').addEventListener('click', () =>
466 {
467     let recursosADevolver = 0;
468     this.gestor.objetosActivos.forEach(obj =>
469     {
470         if (obj.tipo === 'edificio')
471         {
472             recursosADevolver += Math.floor(this.costoEdificio * 0.5);
473         } else if (obj.tipo === 'vehiculo')
474         {
475             recursosADevolver += Math.floor(this.costoVehiculo * 0.5);
476         } else if (obj.tipo === 'ciudadano')
477         {
478             recursosADevolver += Math.floor(this.costoCiudadano * 0.5);
479         }
480     });
481
482     this.gestor.objetosActivos.forEach(obj => {
483         if (obj.tipo === 'edificio')
```

```

484     {
485         this.poolEdificios.liberar(obj);
486     } else if (obj.tipo === 'vehiculo')
487     {
488         this.poolVehiculos.liberar(obj);
489     } else if (obj.tipo === 'ciudadano')
490     {
491         this.poolCiudadanos.liberar(obj);
492     }
493 });
494
495     this.gestor.recursos += recursosADevolver;
496     this.gestor.reciclarTodo();
497     this.gestor.actualizarInterfaz();
498     this.vista.actualizarVisualizacionPool(this.poolEdificios, this.poolVehiculos, this.poolCiudadanos);
499 });
500
501 document.getElementById('testSingleton').addEventListener('click', () =>
502 {
503     this.comprobarSingleton();
504 });
505
506 document.getElementById('guardarEstado').addEventListener('click', () =>
507 {
508     this.mementoEstado = this.gestor.guardarEstado();
509     alert('Estado guardado correctamente');
510 });
511
512 document.getElementById('restaurarEstado').addEventListener('click', () =>
513 {
514     if (this.mementoEstado)
515     {
516         this.gestor.restaurarEstado(this.mementoEstado, this.poolEdificios, this.poolVehiculos, this.poolCiudadanos);
517         this.vista.actualizarVisualizacionPool(this.poolEdificios, this.poolVehiculos, this.poolCiudadanos);
518         alert('Estado restaurado correctamente');
519     }
520     else
521     {
522         alert('No hay estado guardado');
523     }
524 });
525 }
526 }
527
528 const gestorJuego = GestorJuego.obtenerInstancia();
529 const renderizadorDOM = new RenderizadorDOM();
530 const vista = new JuegoVista();
531 gestorJuego.setVista(vista);
532
533 const poolEdificios = new PoolObjetos(
534     () => new Edificio(renderizadorDOM),
535     (obj) => obj.reciclar()

```

```
535     (obj) => obj.reciclar(),
536     5
537 );
538
539 const poolVehiculos = new PoolObjetos(
540     () => new Vehiculo(renderizadorDOM),
541     (obj) => obj.reciclar(),
542     5
543 );
544
545 const poolCiudadanos = new PoolObjetos(
546     () => new Ciudadano(renderizadorDOM),
547     (obj) => obj.reciclar(),
548     5
549 );
550
551 new JuegoControlador(gestorJuego, vista, poolEdificios, poolVehiculos, poolCiudadanos);
552 vista.actualizarVisualizacionPool(poolEdificios, poolVehiculos, poolCiudadanos);
553
554
```

## Examen.css

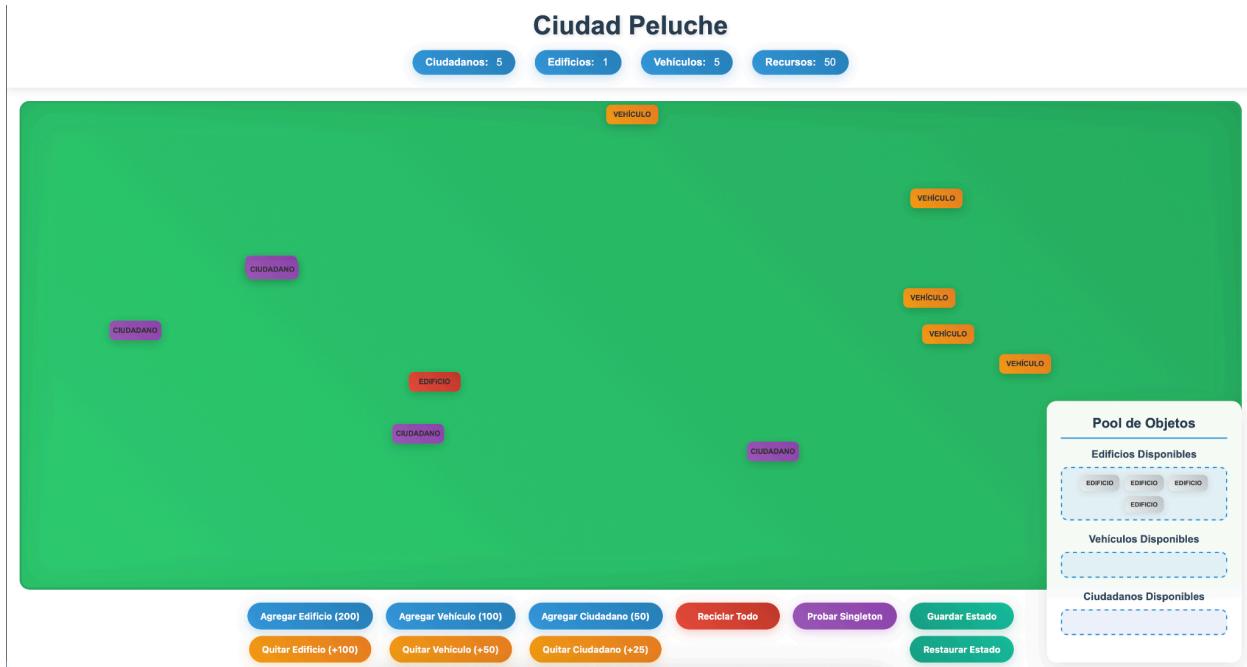
```
1  * {
2      margin: 0;
3      padding: 0;
4      box-sizing: border-box;
5  }
6
7  body {
8      font-family: Arial, sans-serif;
9      background: #fff;
10     min-height: 100vh;
11     color: #333;
12 }
13
14 .contenedor-juego {
15     display: flex;
16     flex-direction: column;
17     height: 100vh;
18     position: relative;
19 }
20
21 .encabezado-juego {
22     background: rgba(255, 255, 255, .95);
23     padding: 20px;
24     box-shadow: 0 2px 10px rgba(0, 0, 0, .1);
25     backdrop-filter: blur(10px);
26 }
27
28 .encabezado-juego h1 {
29     text-align: center;
30     color: #2c3e50;
31     margin-bottom: 15px;
32     font-size: 2.5em;
33     text-shadow: 2px 2px 4px rgba(0, 0, 0, .1);
34 }
35
36 .estadisticas {
37     display: flex;
38     justify-content: center;
39     gap: 30px;
40     flex-wrap: wrap;
41 }
42
43 .estadistica {
44     background: linear-gradient(45deg, #3498db, #2980b9);
45     color: #fff;
46     padding: 10px 20px;
47     border-radius: 25px;
48     box-shadow: 0 4px 15px rgba(52, 152, 219, .3);
49 }
50
51 .etiqueta-estadistica {
52     font-weight: bold;
53     margin-right: 8px;
54 }
55 .area-principal {
56     flex: 1;
```

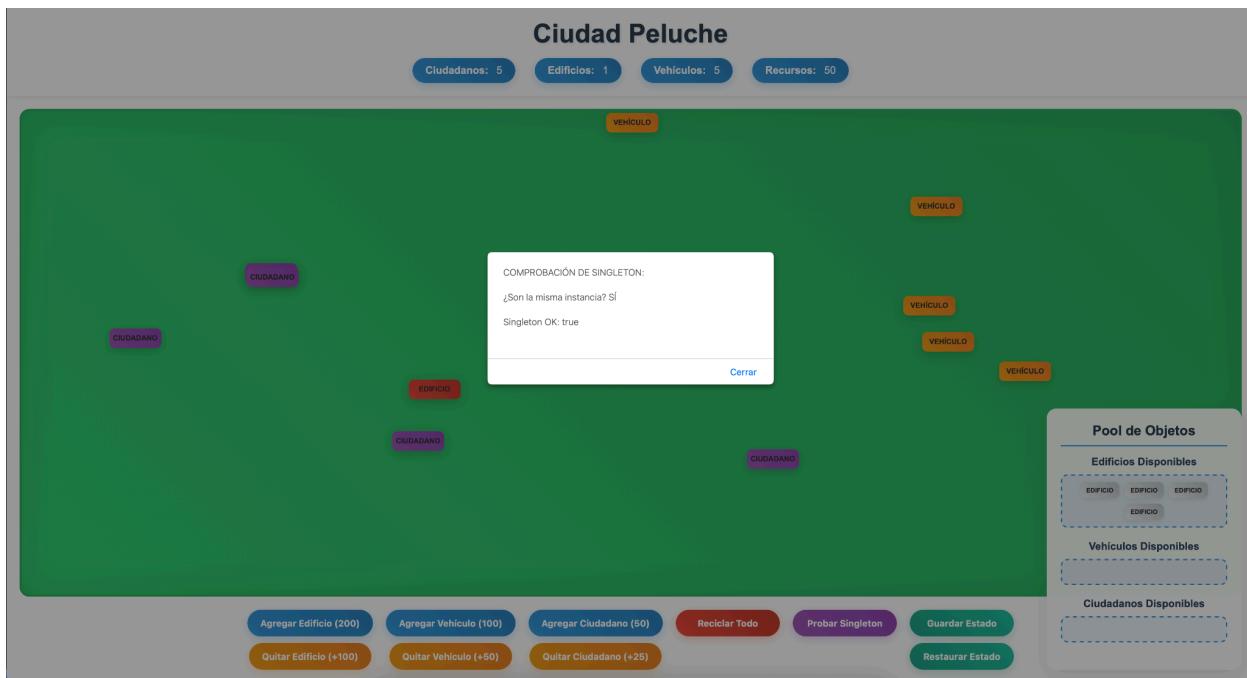
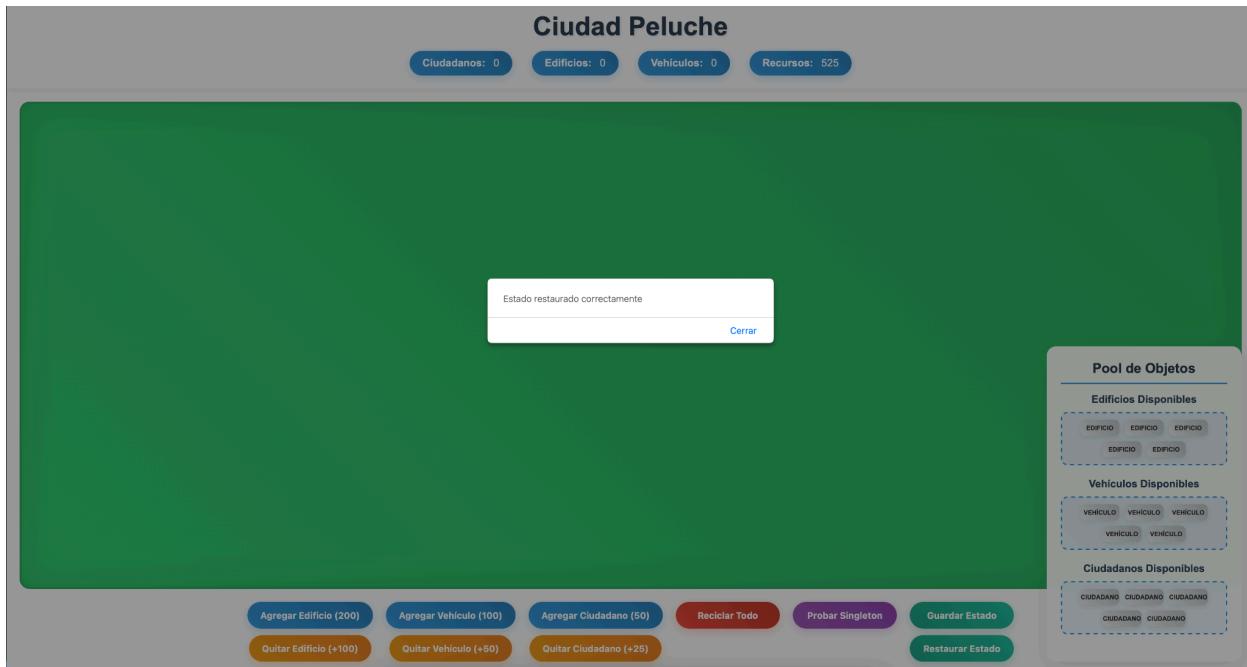
```
57     display: flex;
58     flex-direction: column;
59     padding: 20px;
60     position: relative;
61 }
62
63 .area-ciudad {
64     flex: 1;
65     background: linear-gradient(45deg, #2ecc71, #27ae60);
66     border-radius: 15px;
67     position: relative;
68     overflow: hidden;
69     box-shadow: inset 0 0 50px rgba(0, 0, 0, .1);
70     min-height: 400px;
71     border: 3px solid #27ae60;
72 }
73
74 .edificio, .vehiculo, .ciudadano {
75     position: absolute;
76     width: 80px;
77     height: 30px;
78     display: flex;
79     align-items: center;
80     justify-content: center;
81     font-size: .7em;
82     font-weight: bold;
83     cursor: pointer;
84     border-radius: 8px;
85     box-shadow: 0 4px 15px rgba(0, 0, 0, .2);
86     text-align: center;
87     line-height: 1;
88 }
89
90 .edificio {
91     background: linear-gradient(45deg, #e74c3c, #c0392b);
92 }
93
94 .vehiculo {
95     background: linear-gradient(45deg, #f39c12, #e67e22);
96 }
97
98 .ciudadano {
99     background: linear-gradient(45deg, #9b59b6, #8e44ad);
100 }
101
102 .controles {
103     display: flex;
104     justify-content: center;
105     gap: 20px;
106     margin-top: 20px;
107     flex-wrap: wrap;
108 }
109
110 .grupo-controles {
111     display: flex;
```

```
112     flex-direction: column;
113     gap: 10px;
114     align-items: center;
115   }
116
117   .boton-control {
118     background: linear-gradient(45deg, #3498db, #2980b9);
119     color: #fff;
120     border: none;
121     padding: 12px 20px;
122     border-radius: 25px;
123     cursor: pointer;
124     font-size: 14px;
125     font-weight: bold;
126     box-shadow: 0 4px 15px rgba(52, 152, 219, .3);
127     min-width: 140px;
128   }
129
130   .boton-control.guitar {
131     background: linear-gradient(45deg, #f39c12, #e67e22);
132     box-shadow: 0 4px 15px rgba(243, 156, 18, .3);
133   }
134
135   .boton-control.reciclar {
136     background: linear-gradient(45deg, #e74c3c, #c0392b);
137     box-shadow: 0 4px 15px rgba(231, 76, 60, .3);
138     min-width: 160px;
139   }
140
141   .boton-control.singleton {
142     background: linear-gradient(45deg, #9b59b6, #8e44ad);
143     box-shadow: 0 4px 15px rgba(155, 89, 182, .3);
144     min-width: 160px;
145   }
146
147   .boton-control.memento {
148     background: linear-gradient(45deg, #16a085, #1abc9c);
149     box-shadow: 0 4px 15px rgba(22, 160, 133, .3);
150     min-width: 160px;
151   }
152
153   .panel-pool-objetos {
154     position: fixed;
155     bottom: 20px;
156     right: 20px;
157     width: 300px;
158     background: #rgba(255, 255, 255, .95);
159     border-radius: 15px;
160     padding: 20px;
161     box-shadow: 0 8px 32px rgba(0, 0, 0, .1);
162     backdrop-filter: blur(10px);
163     border: 2px solid #rgba(255, 255, 255, .2);
164     z-index: 1000;
165     max-height: 60vh;
166   }
```

```
167 }
168
169 .panel-pool-objetos h3 {
170   text-align: center;
171   color: #2c3e50;
172   margin-bottom: 15px;
173   font-size: 1.3em;
174   border-bottom: 2px solid #3498db;
175   padding-bottom: 10px;
176 }
177
178 .seccion-pool {
179   margin-bottom: 20px;
180 }
181
182 .seccion-pool h4 {
183   color: #34495e;
184   margin-bottom: 10px;
185   font-size: 1em;
186   text-align: center;
187 }
188
189 .items-pool {
190   display: flex;
191   flex-wrap: wrap;
192   gap: 8px;
193   justify-content: center;
194   min-height: 40px;
195   background: rgba(52, 152, 219, .1);
196   border-radius: 10px;
197   padding: 10px;
198   border: 2px dashed #3498db;
199 }
200
201 .item-pool {
202   width: 60px;
203   height: 25px;
204   display: flex;
205   align-items: center;
206   justify-content: center;
207   background: linear-gradient(45deg, #ecf0f1, #bdc3c7);
208   border-radius: 8px;
209   font-size: .6em;
210   font-weight: bold;
211   box-shadow: 0 2px 8px rgba(0, 0, 0, .1);
212   text-align: center;
213   line-height: 1;
214 }
```

## Ejecución





## Conclusión:

En este examen aprendí a cómo combinar y aplicar la funcionalidad de varios patrones de diseño en un solo sistema, esto al ser algo chico pues es un poco más sencillo, pero en un sistema más grande es fundamental creo yo que se usen estos patrones para tener un estándar claro y tener buenas prácticas como programadores.

