

ASESINO DE MONSTRUOS

Juego VueJS



Funcionamiento

TU

51

MONSTRUO

54



ATACAR

ATAQUE ESPECIAL

CURAR

GIVE UP

EL MONSTRUO LASTIMA AL JUGADOR EN 11

EL JUGADOR GOLPEA DURAMENTE AL MONSTRUO POR 10

EL MONSTRUO LASTIMA AL JUGADOR EN 11

EL JUGADOR GOLPEA AL MONSTRUO POR 3

EL MONSTRUO LASTIMA AL JUGADOR EN 5

EL JUGADOR GOLPEA AL MONSTRUO POR 5

EL MONSTRUO LASTIMA AL JUGADOR EN 5

Diseño



Descargar template inicial

- ✓ Descargar de GITHUB: [\[ESTILOS\]](#)

Analizando el HTML - Botones

```
<section class="row controls">
  <div class="small-12 columns">
    <button id="start-game">EMPEZAR JUEGO NUEVO</button>
  </div>
</section>
<section class="row controls">
  <div class="small-12 columns">
    <button id="attack">ATACAR</button>
    <button id="special-attack">ATAQUE ESPECIAL</button>
    <button id="heal">CURAR</button>
    <button id="give-up">GIVE UP</button>
  </div>
</section>
```

Analizando el HTML – Log de Actividades

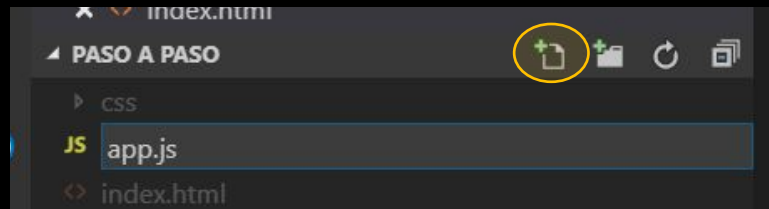
```
<section class="row log">  
  <div class="small-12 columns">  
    <ul>  
      <li>  
  
      </li>  
    </ul>  
  </div>  
</section>
```

Crear la lógica
usando VueJs



Agregar instancia VueJs

1- Crear app.js



2- Importar app.js en el HTML

```
</div>
<script src="app.js"></script>
</body>
</html>
```

← Porque va al final???

3- Crear una nueva instancia de vue

```
<body>
<div id="app">
  <section class="row">
    <div class="small-6 columns">
```

Index.html

```
new Vue({
  el: '#app',
  data: {

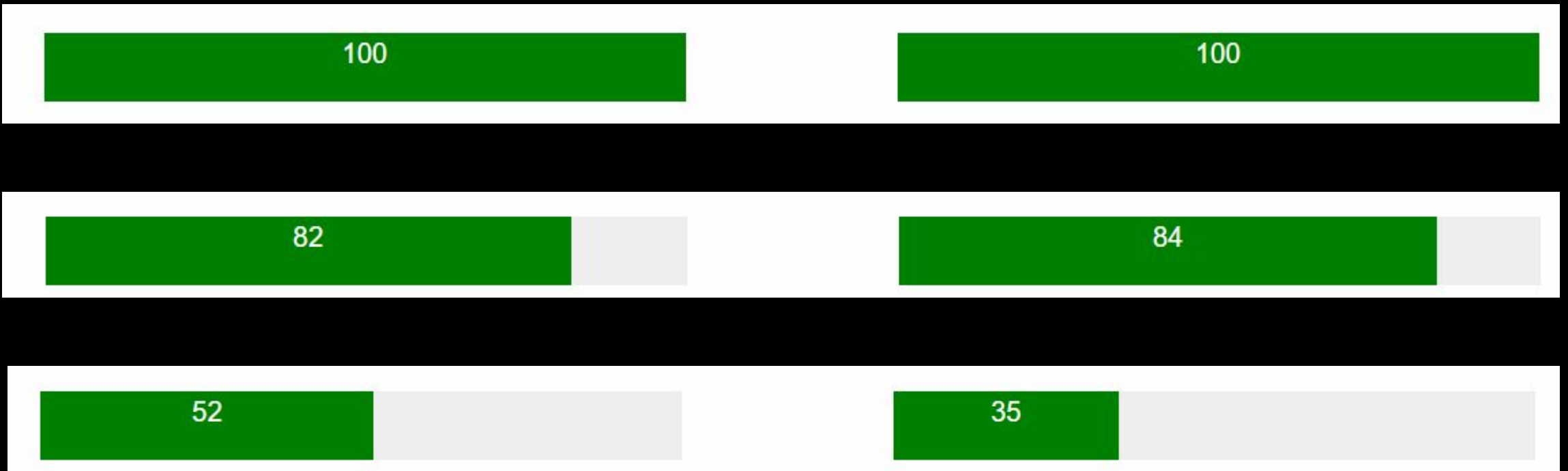
  },
  methods: {

  }
});
```

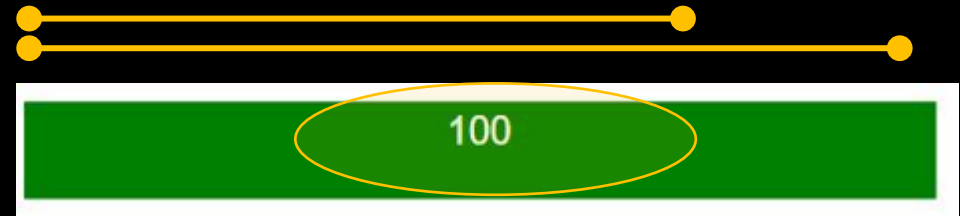
app.js

1er Objetivo: Ajustar el tamaño de la barra

Lograr que las barras de los jugadores tengan un porcentaje y un tamaño acorde a ese porcentaje.



1er Objetivo: Código



```
new Vue({  
  el: 'app',  
  data: {  
    playerHealth: 100,  
    monsterHealth: 100  
  },  
  methods: {  
  
  }  
});
```

app.js

```
<div class="healthbar">  
  <div  
    class="healthbar text-center"  
    style="background-color: green; margin: 0; color: white;"  
    :style="{width: playerHealth + '%'}">  
    {{ playerHealth }}  
  </div>  
</div>  
</div>  
<div class="small-6 columns">  
  <h1 class="text-center">MONSTER</h1>  
  <div class="healthbar">  
    <div  
      class="healthbar text-center"  
      style="background-color: green; margin: 0; color: white;"  
      :style="{width: monsterHealth + '%'}">  
      {{ monsterHealth }}  
    </div>  
  </div>  
</div>
```

index.html

2do Objetivo: Intercambiar visibilidad de botones

El botón EMPEZAR JUEGO NUEVO, debe mostrarse el panel de la jugada.

EMPEZAR JUEGO NUEVO



ATACAR

ATAQUE ESPECIAL

CURAR

GIVE UP

2do Objetivo: Código

```
new Vue({  
  el: 'app',  
  data: {  
    playerHealth: 100,  
    monsterHealth: 100,  
    gameIsRunning: false  
  },  
  methods: {  
  
  }  
});
```

app.js

```
<section class="row controls" v-if="!gameIsRunning">  
  <div class="small-12 columns">  
    <button id="start-game" @click="startGame">  
      EMPEZAR JUEGO NUEVO</button>  
    </div>  
  </section>  
<section class="row controls" v-else>  
  <div class="small-12 columns">  
    <button id="attack" @click="attack">ATACAR</button>  
    <button id="special-attack" @click="specialAttack">ATAQUE ESPECIAL</button>  
    <button id="heal" @click="heal">CURAR</button>  
    <button id="give-up" @click="giveUp">GIVE UP</button>  
  </div>  
</section>
```

v-if

V-else

index.html

Solo porque el tag es de tipo section. Sino usar v-if también.

2do Objetivo - Código

```
<section class="row controls">
  <div class="small-12 columns">
    <button id="start-game" @click="startGame">
      EMPEZAR JUEGO NUEVO</button>
    </div>
  </section>
```

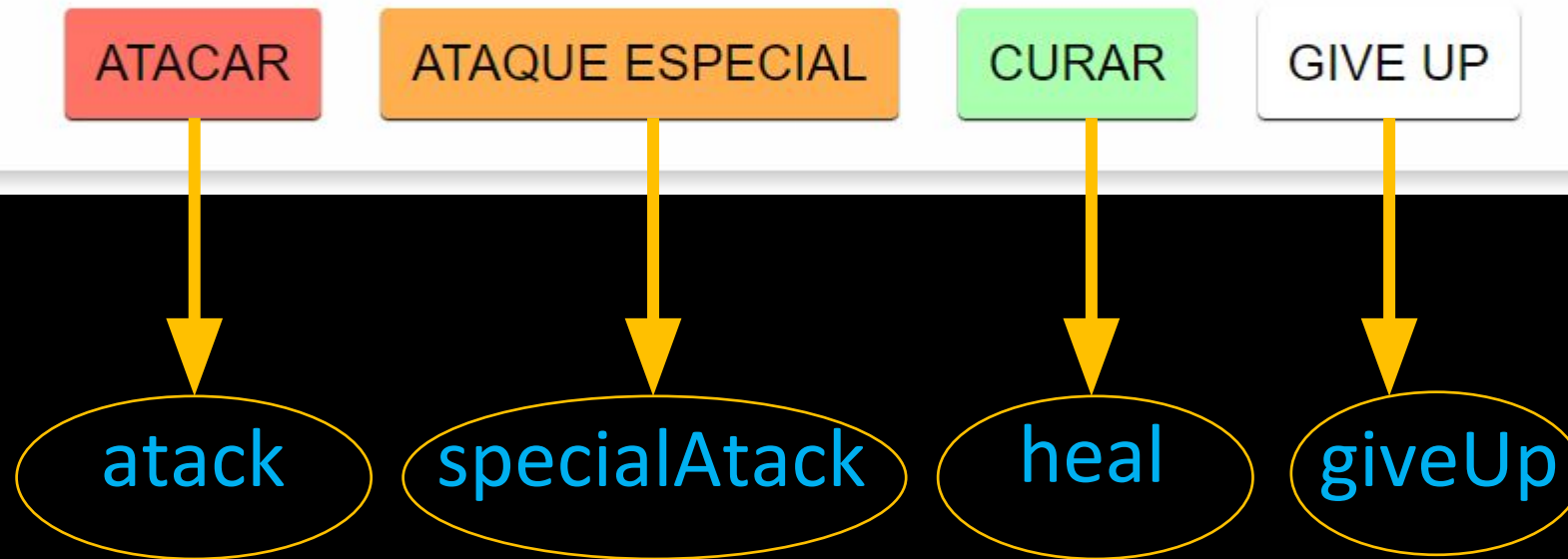
index.html

```
new Vue({
  data: {
    playerHealth: 100,
    monsterHealth: 100,
    gameIsRunning: false
  },
  methods: {
    startGame: function () {
      this.gameIsRunning = true;
      this.playerHealth = 100;
      this.monsterHealth = 100;
    },
  },
});
```

app.js

Eventos Onclick

Llamado a eventos Onclick para escuchar cada vez que el usuario aprieta un botón



Código de eventos

```
<section class="row controls" v-else>
  <div class="small-12 columns">
    <button id="attack" @click="attack">ATACAR</button>
    <button id="special-attack" @click="specialAttack">ATAQUE ESPECIAL</button>
    <button id="heal" @click="heal">CURAR</button>
    <button id="give-up" @click="giveUp">GIVE UP</button>
  </div>
</section>
```

index.html

```
attack: function () {
},
specialAttack: function () {
},
heal: function () {
},
giveUp: function () {
}
```

app.js

```
giveUp: function () {
  this.gameIsRunning = false;
},
```



3er Objetivo - Attack

La lógica que utiliza el juego para atacar se basa en reducir las barras en cierto porcentaje al azar para el monstruo y para el jugador.

- ✓ En el caso del Monstruo, la barra se reduce entre un 3 y 10
- ✓ En el caso del Jugador , la barra se reduce entre 12 y 5

Notar que cada vez que se “Ataca” la barra del jugador tiene más probabilidades de reducirse que la del monstruo.

3er Objetivo – Fórmulas random

3

2

```
Math.max(Math.floor(Math.random() * max) + 1, min);
```

1

MONSTRUO

SIEMPRE

MIN: 5

MAX: 12

JUGADOR

ATACK

MIN: 5

MAX: 12

SPECIALATTAC

K

MIN: 5

MAX: 12

REGLA DEL NEGOCIO: EL MONSTRUO SIEMPRE
ATACA DE LA MISMA FORMA

3er Objetivo – Código attack

app.js

```
attack: function () {  
    var damage = this.calculateDamage(3, 10);  
    this.monsterHealth -= damage;  
      
    if (this.checkWin()) {  
        return;  
    }  
      
    this.monsterAttacks();  
},
```

```
monsterAttacks: function() {  
    var damage = this.calculateDamage(5, 12);  
    this.playerHealth -= damage;  
    this.checkWin();  
},
```

```
calculateDamage: function(min, max) {  
    return Math.max(Math.floor(Math.random() * max) + 1, min);  
},
```

3er Objetivo: Chequear si ganamos o no

app.js

```
checkWin: function() {  
  if (this.monsterHealth <= 0) {  
    if (confirm('Ganaste! Jugar de nuevo?')) {  
      this.startGame();  
    } else {  
      this.gameIsRunning = false;  
    }  
    return true;  
  } else if (this.playerHealth <= 0) {  
    if (confirm('Perdiste! Jugar de nuevo?')) {  
      this.startGame();  
    } else {  
      this.gameIsRunning = false;  
    }  
    return true;  
  }  
  return false;  
}
```

4to Objetivo: SpecialAttack & Heal

SpecialAttack: En este caso, el monstruo pierde más “Energía” que el jugador:

- ✓ En el caso del Monstruo, la barra se reduce entre un 10 y 20
- ✓ En el caso del Jugador , la barra se reduce entre 12 y 5

Heal: En este caso, el jugador recupera más “Energía” que el jugador:

- ✓ En el caso el jugador, recupera 10 de energía
- ✓ Y la barra del jugador se reduce entre 12 y 5

4to Objetivo: Código

app.js

```
specialAttack: function () {  
    var damage = this.calculateDamage(10, 20);  
    this.monsterHealth -= damage;  
  
    if (this.checkWin()) {  
        return;  
    }  
    this.monsterAttacks();  
},
```

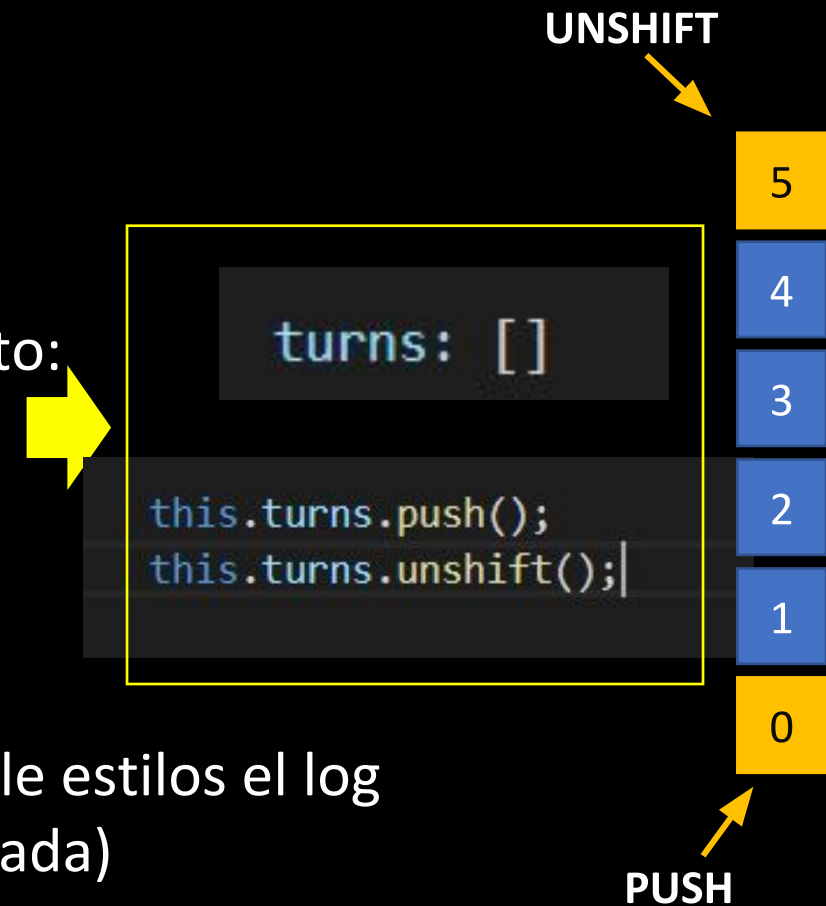
```
heal: function () {  
    if (this.playerHealth <= 90) {  
        this.playerHealth += 10;  
    } else {  
        this.playerHealth = 100;  
    }  
  
    this.monsterAttacks();  
},
```

Añadir el log de
eventos



Estrategia

Creamos un nuevo arreglo de TURNOS que alojará un objeto:



El objeto contiene:

- ✓ Información si es Jugador o monstruo (`isPlayer`) para darle estilos el log
- ✓ Mensaje a emitir en el log (va a depender del tipo de jugada)

```
{
  isPlayer: true,
  text: 'El jugador atacó al monstruo por ' + damage
}
```

```
{
  isPlayer: false,
  text: 'El monstruo atacó al jugador por: ' + damage
}
```


Código de App.js

app.js

```
attack: function () {  
  var damage = this.calculateDamage(3, 10);  
  this.monsterHealth -= damage;  
  this.turns.unshift({  
    isPlayer: true,  
    text: 'El jugador golpea al monstruo por ' + damage  
  });  
  if (this.checkWin()) {  
    return;  
  }  
  this.monsterAttacks();  
},
```


Como harían
SPECIALATTACK y HEAL?

```
monsterAttacks: function() {  
  var damage = this.calculateDamage(5, 12);  
  this.playerHealth -= damage;  
  this.turns.unshift({  
    isPlayer: false,  
    text: 'El monstruo lastima al jugador en ' + damage  
  });  
  this.checkWin();  
},
```

Log en el HTML (sin estilos)

index.html

```
<ul>
  <li v-for="turn in turns">
    {{ turn.text }}
  </li>
</ul>
```



Probando....

1

Que pasa con el log si gano/pierdo y empieza un nuevo juego??

2

Que pasa con el espacio del log cuando aprieto "GIVE UP"??

1

```
startGame: function () {
  this.gameIsRunning = true;
  this.playerHealth = 100;
  this.monsterHealth = 100;
  this.turns = [];
},
```

2


```
<section class="row log" v-if="turns.length > 0">
  <div class="small-12 columns">
    <ul>
      <li v-for="turn in turns">
        {{ turn.text }}
      </li>
    </ul>
  </div>
</section>
```

Darle “formato” al log

```
.log ul .player-turn {  
  color: ■ rgb(0, 140, 255);  
  background-color: ■ #e4e8ff;  
}  
  
.log ul .monster-turn {  
  color: ■ rgb(199, 129, 178);  
  background-color: ■ #ffc0c1;  
}
```

app.css

index.html



```
<ul>  
  <li v-for="turn in turns"  
    :class="{ 'player-turn': turn.isPlayer, 'monster-turn': !turn.isPlayer}">  
    {{ turn.text }}  
  </li>  
</ul>
```

FIN 😊

