

# Informe de Laboratorio 09

## Tema: Angular

Nota

Estudiante(s)	Escuela	Asignatura
Mariel Alisson Jara Mamani mjarama@unsa.edu.pe	Escuela Profesional de Ingeniería de Sistemas	Programación Web 2 Semestre: I Código: 1702122

Laboratorio	Tema	Duración
09	Angular	04 horas

Semestre académico	Fecha de inicio	Fecha de entrega
2023 B	Del 3 Junio 2024	Al 6 Junio 2024

## Laboratorio 09

<b>1 Tarea</b>	<b>3</b>
<b>2 Commits</b>	<b>4</b>
<b>3 Equipos y materiales utilizados</b>	<b>5</b>
<b>4 Ejercicio Propuestos: Juego del ahorcado</b>	<b>6</b>
4.1 Componente app . . . . .	6
4.2 Componente canvas . . . . .	8
4.3 Componente keyboard . . . . .	10
<b>5 Pruebas</b>	<b>12</b>
<b>6 URL del repositorio en GitHub</b>	<b>13</b>
<b>7 Estructura de laboratorio 09</b>	<b>13</b>
<b>8 Rúbrica</b>	<b>14</b>
<b>9 Referencias</b>	<b>14</b>

# 1 Tarea

- **Ejercicio 1:**

- Se desea crear un proyecto con Angular que implemente el juego del ahorcado. Se tendrá un arreglo con posibles palabras a adivinar por parte del usuario. La interfaz la dejamos a gusto de ustedes los programadores.

- **Observaciones:**

- Forme grupos de 2 a 4 personas
- Reportar al profesor que logró culminar la tarea. La tarea debe ser compartida con el profesor en github (CarloCorrales010) y entregada usando el mismo url que se usó para clonar el repositorio. Además cada integrante del grupo deberá crear un video de 5 minutos en flipgrid explicando la aplicación.

## 2 Commits

Se corrige el error del método drawHangman	b2c2ee8			
Alan20 committed 1 minute ago				
Se pone logica en app	a985b95			
Alan20 committed 1 hour ago				
Merge branch 'main' of github.com:Alsnoj20/pw2-24a	b86a633			
Gocardi committed 2 hours ago				
Se importa y se modifica el for del componente Keyboard	e847888			
Alan20 committed 3 hours ago				
Se importa, incluye y estiliza el componente Canvas	55b8c35			
Alan20 committed 3 hours ago				
dependencia rough	f1988c9			
Alan20 committed 3 hours ago				
Merge branch 'main' of https://github.com/Alsnoj20/pw2-24a	e36cbc7			
Alan20 committed 3 hours ago				
implementation of home component	36a1cb9			
Gocardi committed 3 hours ago				
se crea el método dranHangMan para dibujar en el canvas, esto en el componente Canvas	6aef0a3			
Alan20 committed 3 hours ago				
Keyboard component	8dc832f			
JhonatanDziel committed 3 hours ago				
Se agrega los ciclos de vida afterviewinit y onchange en el componente canvas	ac7f2c7			
Alan20 committed 3 hours ago				
Se crea el servicio del juego : HangGame	d8f0c76			
Alan20 committed 4 hours ago				
Se generan los componente Home, Canvas, Game, Keyboard, Results	4dca256			
Alan20 committed 4 hours ago				
Se inicializa el proyecto hangMan con Angular	7d6371a			
Alan20 committed 4 hours ago				
Realización de proyecto sciense-quiz	6f85c12			
Alan20 committed 5 hours ago				
Commits on Jul 3, 2024				
Se crea el component todo-list y todo-list-item	34c2ba8			
Alan20 committed 3 days ago				
Se crea las preguntas	ea9e15f			
Alan20 committed 3 days ago				
Se crea el servicio para manejar preguntas	92f1e44			
Alan20 committed 3 days ago				
Se crean los componentes home, question y results				

Lista de commits realizados en el proyecto.

### 3 Equipos y materiales utilizados

- Cuenta en GitHub con el correo institucional.
- Sistema Operativo Microsoft Windows 10
- Visual Studio Code
- Git
- Windows PowerShell
- Angular
- Navegador Mozilla Firefox

## 4 Ejercicio Propuestos: Juego del ahorcado

Para el siguiente proyecto se ha creado una aplicación con Angular que implementa el juego del ahorcado. Se utiliza un elemento Canvas a través de la API de rough.js para dibujar el ahorcado. Para el diseño, se ha empleado Tailwind CSS. Se generaron los siguientes componentes:

- **app.component:** Componente principal que contiene la lógica del juego.
- **canvas.component:** Componente que contiene el canvas para dibujar el ahorcado.
- **keyboard.component:** Componente que contiene el teclado para seleccionar las letras del juego.

### 4.1 Componente app

Este componente contiene la lógica del juego. Se encarga de seleccionar una palabra aleatoria del arreglo de palabras y de verificar si la letra seleccionada por el usuario se encuentra en la palabra. Además, se encarga de verificar si el usuario ha ganado o perdido el juego.

app.component.ts

```
1 import { Component } from '@angular/core';
2 import { RouterOutlet } from '@angular/router';
3 import { CanvasComponent } from '../canvas/canvas.component';
4 import { KeyboardComponent } from '../keyboard/keyboard.component';
5
6 @Component({
7   selector: 'app-root',
8   standalone: true,
9   imports: [RouterOutlet, CanvasComponent, KeyboardComponent],
10  templateUrl: './app.component.html',
11  styleUrls: ['./app.component.css']
12 })
13 export class AppComponent {
14   WORD:string = '';
15
16   readonly WORD_LIST = [
17     'ANGULAR',
18     'JAVASCRIPT',
19     'TYPESCRIPT',
20     'PROGRAMMING',
21     'DEVELOPMENT',
22     'COMPONENT',
23     'DIRECTIVE',
24     'SERVICE',
25     'MODULE',
26     'ROUTING',
27     'LIFE',
28     'CYCLE',
29     'HTTP',
30     'CLIENT',
31     'SERVER',
32     'DATABASE',
33     'QUERY',
34     'RESPONSE',
35     'REQUEST',
36     'OBSERVABLE',
37     'SUBSCRIPTION',
```

```

38     'OPERATOR',
39     'FUNCTION',
40     'COMPILATION',
41     'TRANSPILATION',
42     'INTERPRETATION',
43 ]
44
45 readonly TITLE = 'Ahorcado';
46 readonly MAX_TRIES = 6;
47 tries_number = 0;
48
49 constructor() {
50     const random = Math.floor(Math.random() * this.WORD_LIST.length);
51     this.WORD = this.WORD_LIST[random];
52 }
53
54 addTries() {
55     this.tries_number += 1;
56
57     if (this.tries_number === this.MAX_TRIES) {
58         alert('Perdiste');
59         this.WORD = this.WORD_LIST[Math.floor(Math.random() * this.WORD_LIST.length)];
60         this.tries_number = 0;
61     }
62 }
63
64 }

```

- Componente principal que contiene la lógica del juego.

app.component.html

```

1 <app-canvas [step]="tries_number"></app-canvas>
2 <app-keyboard
3     [(addTries)]="addTries()"
4     [MAX_TRIES]="MAX_TRIES"
5     [tries_number]="tries_number"
6     [wordPrimary]="WORD"
7 ></app-keyboard>
8 <router-outlet />

```

- Plantilla del componente **app**.
-

## 4.2 Componente canvas

Este componente contiene el canvas para dibujar el ahorcado. Se hace uso de la api rough.js para dibujar las figuras geométricas, esto en función de la variables step pasada como propiedad de entrada @input. A medida que step incrementa, el dibujo del ahorcado se va completando, esto a través de la función drawHangman.

canvas.component.ts

```
1  import { Component, ElementRef, ViewChild, Input, AfterViewInit, OnChanges, SimpleChanges } from '@angular/core'
2
3  // Declaración para rough.js
4  declare var rough: any;
5
6  @Component({
7    selector: 'app-canvas',
8    standalone: true,
9    imports: [],
10   templateUrl: './canvas.component.html',
11   styleUrls: ['./canvas.component.css']
12 })
13
14 export class CanvasComponent implements AfterViewInit, OnChanges {
15   @ViewChild('itemCanvas', { static: false }) canvas: ElementRef<HTMLCanvasElement> | undefined;
16
17   @Input() step: number = 0;
18   private ctx: CanvasRenderingContext2D | null = null;
19   roughCanvas: any;
20
21
22   ngAfterViewInit(): void {
23     if (this.canvas) {
24       this.ctx = this.canvas.nativeElement.getContext('2d');
25       this.roughCanvas = rough.canvas(this.canvas.nativeElement);
26       this.drawHangman(this.step);
27     }
28   }
29
30   ngOnChanges(changes: SimpleChanges): void {
31     if (changes['step'] && !changes['step'].isFirstChange()) {
32       this.drawHangman(this.step);
33     }
34   }
35
36   drawHangman(step: number): void {
37     if (this.canvas && this.ctx) {
38       this.ctx?.clearRect(0, 0, this.canvas?.nativeElement.width, this.canvas?.nativeElement.height);
39
40
41       if (0 <= step){
42         this.roughCanvas.line(50, 450, 200, 450); // Base
43         this.roughCanvas.line(125, 450, 125, 50); // Poste vertical
44         this.roughCanvas.line(125, 50, 280, 50); // Poste horizontal
45         this.roughCanvas.line(280, 50, 280, 100); // Cuerda
46         this.roughCanvas.circle(280, 140, 40); // Cabeza
47       }
48       if (1 <= step){
```



```

49     this.roughCanvas.line(280, 180, 280, 350); // Cuerpo
50 }
51 if (2 <= step) {
52     this.roughCanvas.line(280, 350, 240, 450); // Pierna izquierda
53 }
54
55 if (3 <= step) {
56     this.roughCanvas.line(280, 350, 320, 450); // Pierna derecha
57 }
58 if (4 <= step) {
59     this.roughCanvas.line(280, 280, 220, 200); // Brazo izquierdo
60 }
61 if (5 <= step) {
62     this.roughCanvas.line(280, 280, 340, 200); // Brazo derecho
63 }
64
65 if (6 <= step) {
66     this.roughCanvas.circle(280, 140, 40); // Cabeza
67     this.roughCanvas.line(270, 130, 260, 120); // Ojo derecho
68     this.roughCanvas.line(260, 130, 270, 120);
69     this.roughCanvas.line(290, 130, 300, 120); // Ojo izquierdo
70     this.roughCanvas.line(300, 130, 290, 120);
71     this.roughCanvas.line(270, 150, 300, 145, { roughness: 4 }); // Boca
72     this.roughCanvas.arc(280, 148, 20, 10, 0, 0.6 * Math.PI, true); // Lengua
73 }
74 }
75 }
76 }

```

- Componente para el canvas.

canvas.component.html

```

1 <div class="w-full max-w-md bg-white rounded-md xd">
2   <canvas #itemCanvas width="370" height="490"></canvas>
3 </div>

```

- Plantilla del componente **canvas**.
- La hoja de estilo esta aqui mismo por taiwlind.

### 4.3 Componente keyboard

Este componente manda una letra. Se hace uso de la función selectLetter para seleccionar la letra y verificar si esta se encuentra en la palabra. Además, se verifica si el usuario ha ganado o perdido el juego.

keyboard.component.ts

```
1  import { Component, Input, Output, EventEmitter, OnInit } from '@angular/core';
2
3  @Component({
4    selector: 'app-keyboard',
5    standalone: true,
6    imports: [],
7    templateUrl: './keyboard.component.html',
8  })
9  export class KeyboardComponent implements OnInit {
10    @Input() wordPrimary = '';
11    @Input() MAX_TRIES = 6;
12    @Input() tries_number = 0;
13
14    @Output() addTries = new EventEmitter<void>();
15
16    word: string[] = [];
17    userWord: string[] = [];
18
19    userLetter: string;
20
21    tries: string[] = [];
22
23    ngOnInit(): void {
24      this.word = this.wordPrimary.toUpperCase().split('');
25      this.userWord = this.word.map(() => ' ');
26    }
27
28    constructor() {
29      this.userLetter = '';
30    }
31
32    changeLetter(event: Event) {
33      this.userLetter = (event.target as HTMLInputElement).value;
34    }
35
36    userTry() {
37      const letter = this.userLetter.toUpperCase();
38      if (this.tries.includes(letter)) {
39        return;
40      }
41
42      this.tries.push(letter);
43
44      if (!this.word.includes(letter)) {
45        this.addTries.emit();
46
47        this.userLetter = '';
48        return;
49      }
50    }
51  }
```

```
50
51     this.word.forEach((l, i) => {
52         if (l === letter) {
53             this.userWord[i] = letter;
54         }
55     });
56
57     this.userLetter = '';
58 }
59 }
```

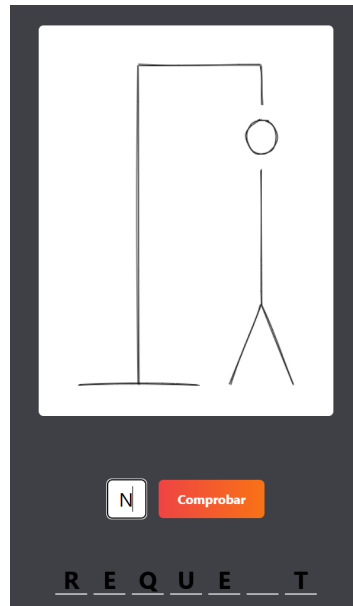
- Componente para el teclado.

keyboard.component.html

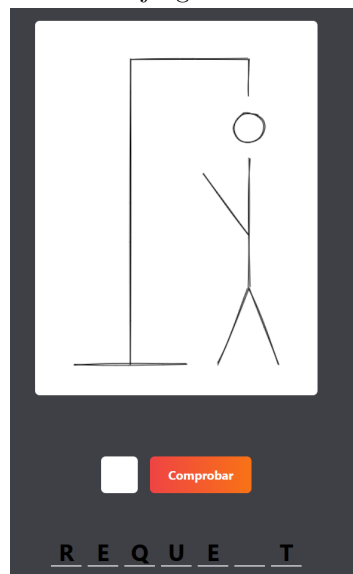
```
1 <div class="flex flex-col gap-10 mt-20 text-black">
2   <div class="flex justify-center gap-4">
3     <input
4       type="text"
5       name="letter"
6       id="letter"
7       [maxLength]="1"
8       class="w-12 px-4 py-2 rounded-md text-2xl"
9       value="{{userLetter}}"
10      [(change)]="changeLetter($event)"
11    >
12    <button class="bg-gradient-to-r from-red-500 to-orange-500 px-6 py-3 rounded-md font-bold text-white hover:s
13      Comprobar
14    </button>
15  </div>
16
17  <div class="flex justify-center gap-2">
18    @for (letter of userWord; track $index) {
19      <span class="w-10 border-b-2 h-14 text-center text-3xl flex justify-center items-end font-bold">
20        {{letter}}
21      </span>
22    }
23  </div>
24 </div>
```

- Plantilla del componente **keyboard**.

## 5 Pruebas



Prueba del juego del ahorcado.



Prueba 2 del juego del ahorcado.

## 6 URL del repositorio en GitHub

- <https://github.com/Alsnj20/pw2-24a/tree/main/lab09>

## 7 Estructura de laboratorio 09

- El contenido que se entrega en este laboratorio es el siguiente:

```
lab09/
|--Angular/
|  |--hangMan/
|    |--src/
|      |--app/
|        |--app.component.ts
|        |--app.component.html
|        |--app.component.css
|      |--canvas/
|        |--canvas.component.ts
|        |--canvas.component.html
|      |--keyboard/
|        |--keyboard.component.ts
|        |--keyboard.component.html
|        |--keyboard.component.css
|--Latex/
|  |--linopinto_pw2_24a_lab09.tex
|  |--linopinto_pw2_24a_lab09.pdf
|--img/
|  |--commits.png
|  |--prueba.png
|  |--prueba2.png
|--science-quiz/
--gitignore
```

## 8 Rúbrica

Tabla: Rúbrica para contenido del Informe y evidencias

Contenido y demostración		Puntos	Checklist	Estudiante	Profesor
1. GitHub	Repositorio se pudo clonar y se evidencia la estructura adecuada para revisar los entregables. (Se descontará puntos por error o observación)	4	×	4	
2. Commits	Hay porciones de código fuente asociado a los commits planificados con explicaciones detalladas. (El profesor puede preguntar para refrendar calificación)	4	×	4	
3. Ejecución	Se incluyen comandos para ejecuciones y pruebas del código fuente explicadas gradualmente que permitirían replicar el proyecto. (Se descontará puntos por cada omisión)	4	×	4	
4. Pregunta	Se responde con completitud a la pregunta formulada en la tarea. (El profesor puede preguntar para refrendar calificación)	2	×	2	
7.Ortografía	El documento no muestra errores ortográficos. (Se descontará puntos por error encontrado)	2	×	1	
8. Madurez	El Informe muestra de manera general una evolución de la madurez del código fuente con explicaciones puntuales pero precisas, agregando diagramas generados a partir del código fuente y refleja un acabado impecable. (El profesor puede preguntar para refrendar calificación)	4	×	4	
<b>Total</b>		20	Completo	19	

## 9 Referencias

- <https://github.com/>
- <https://git-scm.com/>
- <https://www.w3schools.com/python/>