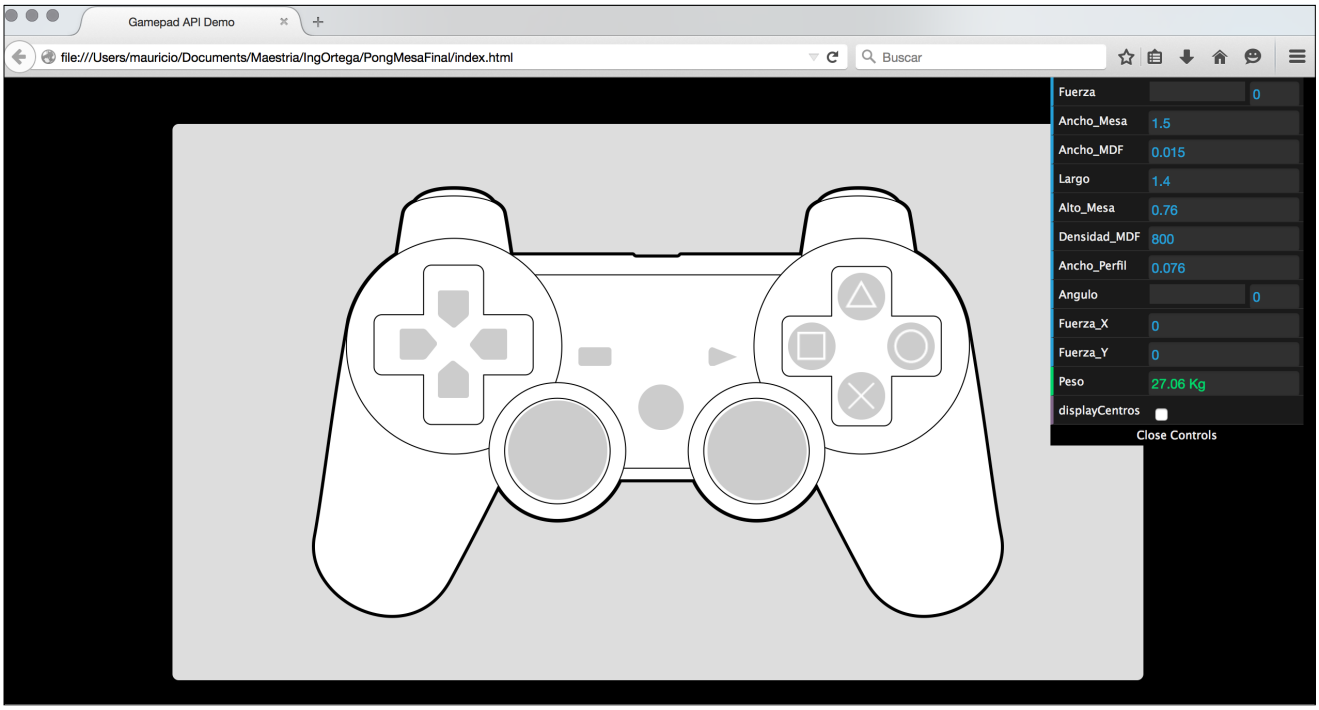
Documentación API, simulación Pong.



Mauricio Duque Orozco
Fredy Llulluna Llumiquinga
Mayo de 2015.
API mesa de Pong.

DOCUMENTACIÓN API

Objetivo

Con el presente informe, se pretende documentar las funcionalidad de la API para la mesa de PONG, con la finalidad de facilitar su estudio, manejo y futuras mejoras.

Objetivos específicos.

I. Documentar de manera precisa las funcionalidad que ofrece la API.

II. Incentivar futuros trabajos de investigación, en la simulación de mecanismos.

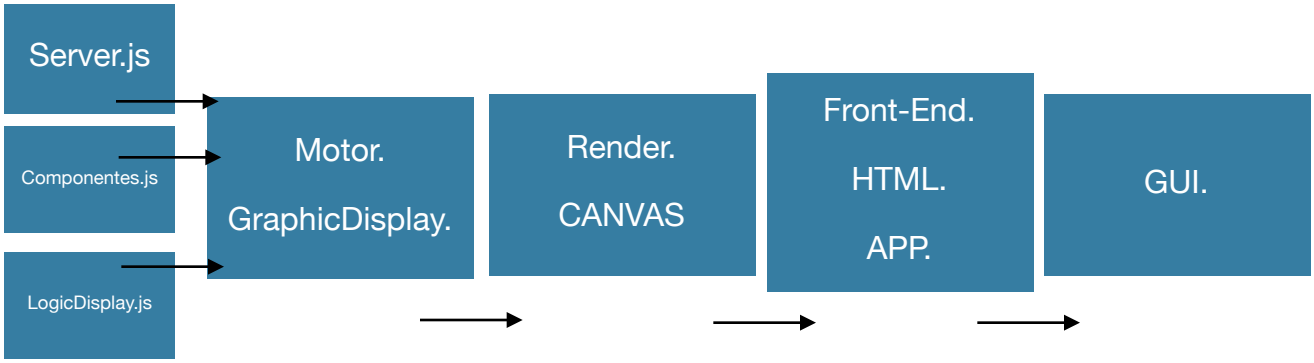
III. Crear una plataforma escalable, para la simulación de mecanismos que permitan analizar de manera rápida y optima los diferentes sistemas.

Documentación.

La API de pong, es una versión beta, para uso académico, con la finalidad de permitir el análisis de simulación de sistemas mecánicos eslabonados, en esta primera versión se enfoca principalmente en la simulación de una mesa de ping-pong, a travez de un mecanismo paraleloide.

Estructura.

La API cuenta con una estructura modular. como lo muestra la siguiente gráfica.



Server.js

Esta clase proporciona un lienzo de trabajo llamado ‘canvas’ fijando los datos de:

a. Tamaño: Alto y Ancho, en pixeles.

b. Id: nombre del elemento html.

c. Crea el contexto: especificamente ‘2D’.

d. Instancia un nuevo elemento de ‘GraphicDisplay’ y de ‘FizzyText’

e. Captura los datos obtenidos por el ‘FizzyText’ y los pone a disposición del elemento ‘gd’.

f. Captura los evenetos de ‘FizzyText’.

g. Dispone un intervalo de 5 milisegundos para ejecutar la funcionalidad ‘gd.execute’ y ‘update’.

Componentes.js.

Se crea un modelo de componentes, genéricos para estandarizar la construcción de elementos.

1. Círculos.
2. Linea.
3. Linea con parametros.
4. Rectángulos.

* **Line**(punto 1 en ‘x’, punto 1 en ‘y’, punto 2 en ‘x’, punto 2 en ‘y’, ‘nombre del elemento', ‘color del elemento', grueso de linea).
* **LineAngle**(punto 1 en ‘x’, punto 1 en ‘y’, longitud de linea, angulo en grados, ‘nombre del elemento', ‘color del elemento', grueso de linea).
* **Rectangle**(punto 1 en ‘x’, punto 1 en ‘y’, punto 2 en ‘x’, punto 2 en ‘y’, ‘nombre del elemento', ‘color del elemento', grueso de linea).
* **Circle**(punto 1 en ‘x’, punto 1 en ‘y’, radio).

Posee dos funcionalidades:

1. Component.prototype.setActive = function(active) {

    this.active = active;

};

2. Component.prototype.isActive = function() {

    return this.active;

};

Ambas funcionalidades son: ‘get’ y ‘set’ de activación del elemento, permitiendo su visualization o no.

Asignación de de tipo por elemento, dependiendo de su constructor.

LogicDisplay.js

Elemento constructor de arreglo para almacenar los componentes creados a partir del la estructura 'Componentes.js', la finalidad es poder gráficar los elementos de manera dinámica, y administrarlos de mejor manera, dicha administración se emplea en el elemento 'GraphicDisplay.js'.

GraphicDisplay.js

Básicamente es el motor de la API, esta se encarga de la gestión de renderizado y de los cálculos necesarios para la simulación del mecanismo, cuenta con las siguientes funciones:

(ctx, width, height) => contexto del lienzo de trabajo, con las dimensiones del canvas, instancia variables a usar.

a. init.

* Crea nuevas instancias del constructor de 'LogicDisplay' => this.logicDisplay.
* Normaliza las medidas, es decir llama a la funcionalidad para que aplique los factores de escala, sobre las variables.
* Llama la creación de los elementos básicos.
* Llama la funcionalidad de el calculo de los centros de masa.
* Llama la función para creación de ejes.

Mandos básicos de arranque, solo se ejecuta una vez.

b. normMedidas.

Aplica los factores de escala a cada una de las medidas.

c. elements.

Crea los elementos básicos a partir de las medidas.

d. centrosMasa. (components)

Calcula los centros de masa de los componentes.

e. ejes. (components)

Calcula los ejes de los componentes.

f. clearAll.

Borra los componentes.

g. drawLineAngle. (x , y , l , a , color, lineWidth)

Dibuja lineas sobre el contexto a partir del arreglo suministrado.

h. drawLine. (x , y , x1 , y1, color, lineWidth)

Dibuja lineas sobre el contexto a partir del arreglo suministrado.

i. drawRectangle. (x,y,x1,y1,color, lineWidth)

Dibuja Rectángulos sobre el contexto a partir del arreglo suministrado.

j. drawCircle. (x1, y1, radius)

Dibuja Círculos sobre el contexto a partir del arreglo suministrado.

k. execute.

La funcionalidad que se encarga de mover la dinamica de la aplicación, es la que se ejecuta cada 5 milisegundos, encargando del renderizado de los componentes.

l. drawAllComponents. (components)

Dibuja todos los componentes.

m. drawComponent. (component)

Dibuja componente por componente.

n. move. (components)

Movimientos relativos de los elementos, hace los cálculos necesarios para el renderizado.

o. findObject. (components,name) => return i

Encuentra los componentes a partir de su nombre y retorna su posición en el arreglo.

p. setAngle. (angle)

Fija el angulo principal.

q. getAngle. (angle) => return theta

Obtiene el angulo en radianes.

r. getCoorX. (x , a , l) => return desp_X

Obtiene la coordenada en X del elemento.

s. getCoorY. (y , a , l) => return desp_Y

Obtiene la coordenada en Y del elemento.

t. setFuerza. (fuerza)

Fija la fuerza aplicada.

u. getDistance. (x,y,x1,y1) => return distance.toFixed(2)

Obtiene la distancia entre dos puntos suministrados.

v. setDisplay. (value)

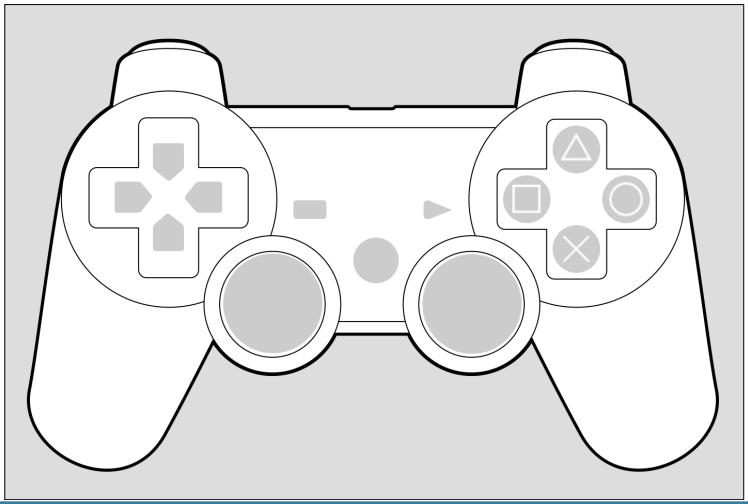
Determina la visualización o no del elemento.

PANEL DE CONTROL

Básicamente el panel de control con el que se dispone es muy sencillo, se disponen las medidas requeridas por el sistema. Se puede visualizar el cambio de ángulo y la fuerza realizada en 'X' y en 'Y'; también se puede variar la fuerza requerida para mover el sistema; por ultimo esta la opción para la visualización de los elementos, solo con el 'check'.

USO DEL GAMEPAD.

También se puede simular el movimiento de la mesa mediante el uso de un mando de PS3, esta opción es automática y detecta el control, si esta habilitado o no; esta configurados todos los botones, pero para términos de la simulación, solo se usa el joystick izquierdo en el eje 'Y' positivo.



VISUALIZACIÓN DE LA SIMULACIÓN.

En la visualización de la simulación, se observa la acción de las fuerzas que actúan en el sistema, y el movimiento de cada uno de los componentes, cabe recalcar que los cálculos son realizados de manera automática de acuerdo a las dimensiones establecidas, el código es recursivo y adaptativo a cualquier estructura similar, que se comporte de la misma manera.