Librerías JavaScript – Processing.js

Programación Multimedia.

G.I.M.

Francisco Grimaldo, Inmaculada Coma

Índice

- Librerías JavaScript
- Processing.js vs Three.js
- Processing & Processing.js
- Plantillas de uso básicas
- Interacción JavaScript – Processing
- Paso de datos entre JavaScript y Processing

Librerías JavaScript

- HTML5 + <canvas>:
  - Introducción de contenido 2D y 3D interactivo.
  - Contextos “2d” y “webgl” permiten el desarrollo de gráficos acelerados por hardware
- Librerías gráficas JavaScript:
  - Reducción del tiempo de desarrollo.
  - Incluyen funcionalidades útiles:
    - Renderizado, física, detección de colisiones, sonido, animación, inteligencia artificial, redes, etc.
  - Catálogo muy amplio de posibilidades:
    - http://jster.net/

Three.js vs Processing.js (1/2)

- Motor 3D ligero.
- Proporciona:
  - API para construcción del grafo de escena.
  - Objetos comunes utilizados por los profesionales de programación gráfica (p. ej. Materiales, iluminación, texturas…)
- Aplicación principal:
  - Renderizado de escenas 3D complejas mediante JavaScript.

Three.js vs Processing.js (2/2)

- Interfaz web al lenguaje de programación visual Processing.
- Usado por:
  - Artistas, diseñadores e investigadores.
  - Aprendices en programación procedural multimedia.
- Usado para:
  - Arte digital.
  - Animaciones interactivas.
  - Visualización de datos (p.ej. http://feltron.com).

Processing

- http://www.processing.org
- Lenguaje e IDE open-source para desarrollo de arte digital.
- Iniciado en 2001 en el MIT Media Lab.
- Crea “sketches” con sintaxis similar a Java.
- Se basa en Java, pero simplifica la sintaxis y el modelo de programación gráfica.
- Integración web inicial mediante applets.

Processing.js

- http://processingjs.org
- Traduce Processing a JavaScript.
- Usa <canvas> de HTML5.
- Es rápido, sencillo y utiliza WebGL para 3D.
- Inicia en 2008 por John Reising y mantenido hoy por estudiantes del Seneca College.
- Soportado por: Firefox, Opera, IE9, Safari y Chrome.

Plantillas de uso básicas (1/4)

- Código HTML:

```
<!DOCTYPE html>
<html>
<head>
    <title>Hello Web - Processing.js Test</title>
    <script src="processing-1.4.1.min.js"></script>
</head>
<body>
    <h1>Processing.js Test</h1>
    <p>This is my first Processing.js web-based sketch:</p>
    <canvas data-processing-sources="hello-web.pde"></canvas>
</body>
</html>
```

Plantillas de uso básicas (2/4)

- Código Processing (*.pde):

```
// Variables globales
int i = 0;

// Inicialización
void setup() {
    size(200, 200);
    background(255);
    smooth();
    strokeWeight(15);
    frameRate(24);
}
```

```
// Función de dibujado
void draw() {
    stroke(random(50),
           random(255),
           random(255),
           100);
    line(i, 0, random(0, width), height);
    if (i < width) {
        i++;
    } else {
        i = 0;
    }
}
```

Plantillas de uso básicas (3/4)

- Incluir código Processing en HTML:

```
<script src="processing-1.4.1.min.js"></script>
<script type="text/processing" data-processing-target="mycanvas">
void setup() {
  size(200,200);
  background(125);
  fill(255);
  noLoop();
  PFont fontA = loadFont("courier");
  textFont(fontA, 14);
}
void draw() {
  text("Hello Web!",20,20);
  println("Hello ErrorLog!");
}
</script>
<canvas id="mycanvas"></canvas>
```

Plantillas de uso básicas (4/4)

- Crear la escena con el API JS de Processing:

```
<script src="processing-1.4.1-api.min.js"></script>
<canvas id="canvas1"></canvas>
<script type="application/javascript">
    function sketchProc(processing) {
     processing.setup = function() {
       processing.size(200, 200);
       processing.background(100);
       processing.stroke(255);
       processing.ellipse(50, 50, 25, 25);
       processing.println("hello web!");
     }
    }
    var canvas = document.getElementById("canvas1");
    var processingInstance = new Processing(canvas, sketchProc);
</script>
```

Interacción JavaScript-Processing (1/2)

```
<!DOCTYPE html>
<html>
  <head>
    <meta charset="utf-8">
    <title>My Processing Page</title>
    <script type="text/javascript" src="processing-1.4.1.min.js"></script>
    <script type="text/javascript">
    function drawSomeText(id) {
      var pjs = Processing.getInstanceById(id);
      var text = document.getElementById('inputtext').value;
      pjs.drawText(text); }
    </script>
  </head>
  <body>
    <canvas id="prueba" data-processing-sources="js-pc5.pde"></canvas>
    <input type="textfield" value="my text" id="inputtext">
    <button type="button" onclick="drawSomeText('prueba')">Write!</button>
  </body>
</html>
```

Interacción JavaScript-Processing (2/2)

- Función en Processing llamada por JS:

```
void setup() {
    size(200,200);
    // No es necesario refrescar
    noLoop();
    stroke(#FFEE88);
    fill(#FFEE88);
    background(#000033);
    textSize(24);
}
```

```
void drawText(String t)
{
    background(#000033);
    // Obtener la longitud del texto
    float twidth = textWidth(t);
    // Escribir el texto centrado
    text(t, (width - twidth)/2, height/2);
}
```

Interacción Processing-JavaScript (1/2)

```
<!DOCTYPE html>
<html>
  <head>
    <meta charset="utf-8">
    <title>My Processing Page</title>
    <script type="text/javascript" src="processing-1.4.1.min.js"></script>
    <script type="text/javascript">
    function showXYCoordinates(x, y) {
      document.getElementById('xcoord').value = x;
      document.getElementById('ycoord').value = y;
    }
    </script>
  </head>
  <body>
    <canvas id="prueba" data-processing-sources="pc5-js.pde"></canvas>
    X:<input type="textfield" id="xcoord" >
    Y:<input type="textfield" id="ycoord" >
  </body>
</html>
```

Interacción Processing-JavaScript (2/2)

- Llamada de Processing a JS:

```
void setup() {
   size(200,200);
   stroke(255);
   background(0);
   noLoop();
}

void draw() {
   fill(0,0,0,40);
   rect(-1,-1,width+2,height+2);
}
```

```
void mouseMoved() {
   line(mouseX,0,mouseX,height);
   line(0,mouseY,width,mouseY);
   redraw();

   showXYCoordinates(mouseX, mouseY);

}
```

- El parser de Pjs ignora el código JS.
- Para compatibilidad con Processing, usar interfaces.

Paso datos JavaScript-Processing

- Carga de datos desde JS:
  - Crear objetos de clases definidas en Processing
  - http://processingjs.org/articles/PomaxGuide.html#pobj
- Carga de datos desde Processing:
  - Desde ficheros JSON:
  - http://processingjs.org/articles/PomaxGuide.html#json
  - Desde ficheros XML:
  - http://processingjs.org/articles/PomaxGuide.html#xml
  - Desde ficheros SVG:
  - http://processingjs.org/articles/PomaxGuide.html#svg

Referencias Processing.js

- ***JSter:*** Catálogo con más de 1000 librerías en JavaScript y herramientas para el desarrollo web.
  - http://jster.net/
- ***Processing:*** Sitio oficial con gran cantidad de documentación, ejemplos y recursos descargables.
  - http://www.processing.org
  - http://processing.org/reference
- ***Processing.js:*** Sitio oficial con tutoriales de referencia, ejemplos de aplicación y recursos descargables.
  - http://processingjs.org
  - http://processingjs.org/learning/
  - http://processingjs.org/articles/PomaxGuide.html