Telerik Academy

# SVG
# Scalable Vector Graphics

ass

Antony Jekov

Academy Intern

http://jekov.telerik-students.org

Telerik Software Academy

http://academy.telerik.com

# Table of Contents

- SVG Overview
- Vector Graphics Overview
- Simple SVG Elements
- Animations and Filters
- SVG Libraries
  - Raphael.js
- SVG Generating Tools

# SVG Overview

- SVG is a language for describing two dimensional vector graphics using XML.
- SVG stands for Scalable Vector Graphics
- SVG is platform independent and is freely understood by most browsers, where versions of Internet Explorer 8 and bellow need an additional plugin.

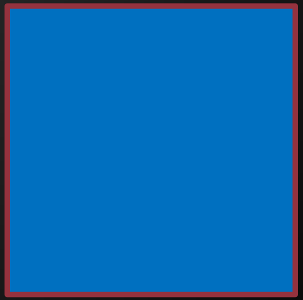
# Vector Graphics Overview

- Based on mathematical expressions
- Consist of geometrical primitives such as points, lines, curves and shapes or polygons.
- Used to represent images in computer graphics.
- Vectors are locations called paths or strokes and they represent a point in two dimensional space that holds coordinates for the ‘x’ and ‘y’ axis.

# Using SVG in a web page

- To use SVG you need to simply open the <svg> tag and to start defining your shapes using xml notation.

```
<svg>
   <rect width="150" height="150" fill="blue“
       stroke="red" stroke-width="2" />
</svg>
```



# Simple SVG

Live Demo

# Simple Animations

- Simple animations can be performed by adding <animate> tag with some additional properties.

```
<circle cx="200" cy="200" r="100" fill="red">
   <animate attributeName="fill" to="blue" dur=".4s"
      repeatCount="indefinite" />
</circle>
<rect width="50" height="50" x="600" fill="red">
   <animate attributeName="y" from="-50" to="600" dur="5s"
      repeatCount="indefinite" />
            </rect>
<ellipse cx="200" cy="400" rx="50" ry="75" fill="purple">
   <animate attributeName="rx" to="75" dur="4s"
      repeatCount="indefinite" />
</ellipse>
```

# Animations Demo

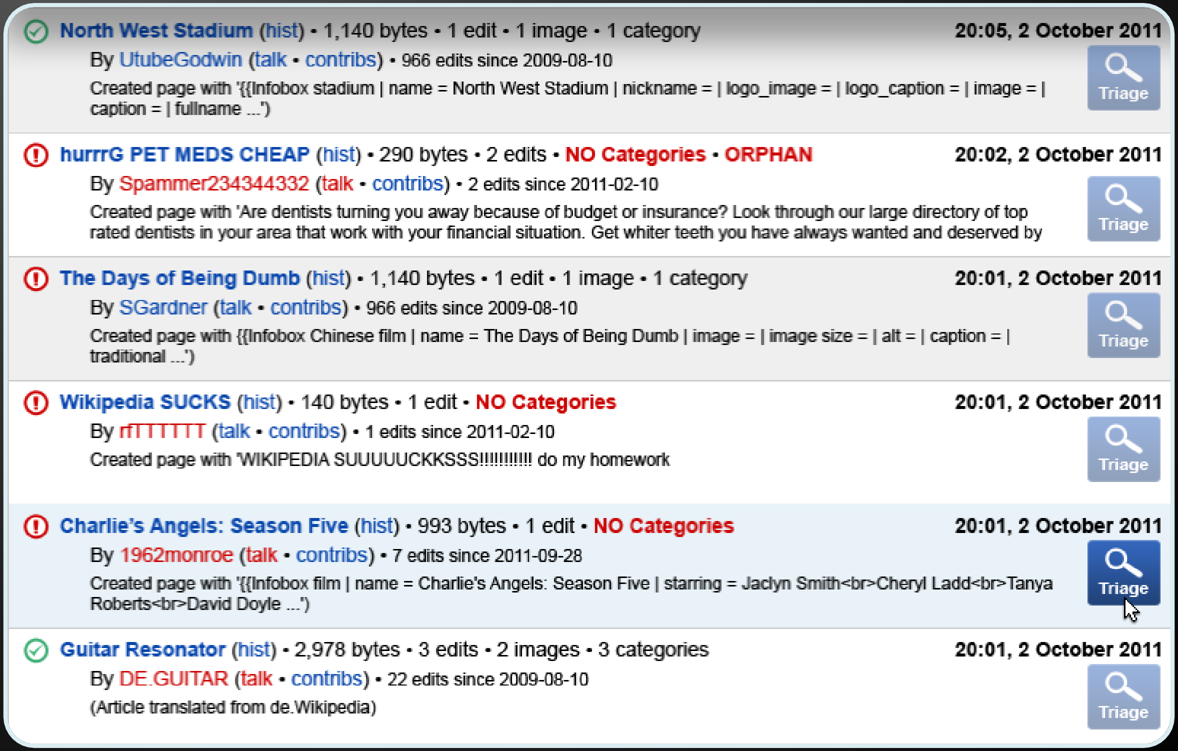
Live Demo

# Filters

- **SVG filters**

# List Sections

## Live Demo



✓ **North West Stadium** (hist) • 1,140 bytes • 1 edit • 1 image • 1 category **20:05, 2 October 2011**
By UtubeGodwin (talk • contribs) • 966 edits since 2009-08-10
Created page with '{{Infobox stadium | name = North West Stadium | nickname = | logo_image = | logo_caption = | image = | caption = | fullname ...')

! **hurrrG PET MEDS CHEAP** (hist) • 290 bytes • 2 edits • **NO Categories** • **ORPHAN** **20:02, 2 October 2011**
By Spammer234344332 (talk • contribs) • 2 edits since 2011-02-10
Created page with 'Are dentists turning you away because of budget or insurance? Look through our large directory of top rated dentists in your area that work with your financial situation. Get whiter teeth you have always wanted and deserved by

! **The Days of Being Dumb** (hist) • 1,140 bytes • 1 edit • 1 image • 1 category **20:01, 2 October 2011**
By SGardner (talk • contribs) • 966 edits since 2009-08-10
Created page with {{Infobox Chinese film | name = The Days of Being Dumb | image = | image size = | alt = | caption = | traditional ...')

! **Wikipedia SUCKS** (hist) • 140 bytes • 1 edit • **NO Categories** **20:01, 2 October 2011**
By rfTTTTTT (talk • contribs) • 1 edits since 2011-02-10
Created page with 'WIKIPEDIA SUUUUUCKKSSS!!!!!!!!!!! do my homework

! **Charlie's Angels: Season Five** (hist) • 993 bytes • 1 edit • **NO Categories** **20:01, 2 October 2011**
By 1962monroe (talk • contribs) • 7 edits since 2011-09-28
Created page with '{{Infobox film | name = Charlie's Angels: Season Five | starring = Jaclyn Smith<br>Cheryl Ladd<br>Tanya Roberts<br>David Doyle ...')

✓ **Guitar Resonator** (hist) • 2,978 bytes • 3 edits • 2 images • 3 categories **20:01, 2 October 2011**
By DE.GUITAR (talk • contribs) • 22 edits since 2009-08-10
(Article translated from de.Wikipedia)

# Inverted Sections

- As said earlier, sections work like a if-else statement
  - The regular sections are the IF statements
  - The inverted sections are the ELSE statements
- Inverted sections start with `{{^prop}}` and end with `{{/prop}}`
- Inverted sections are rendered only if the property is evaluated to "false"

# Inverted Sections: Example

- Only one of the students has no marks

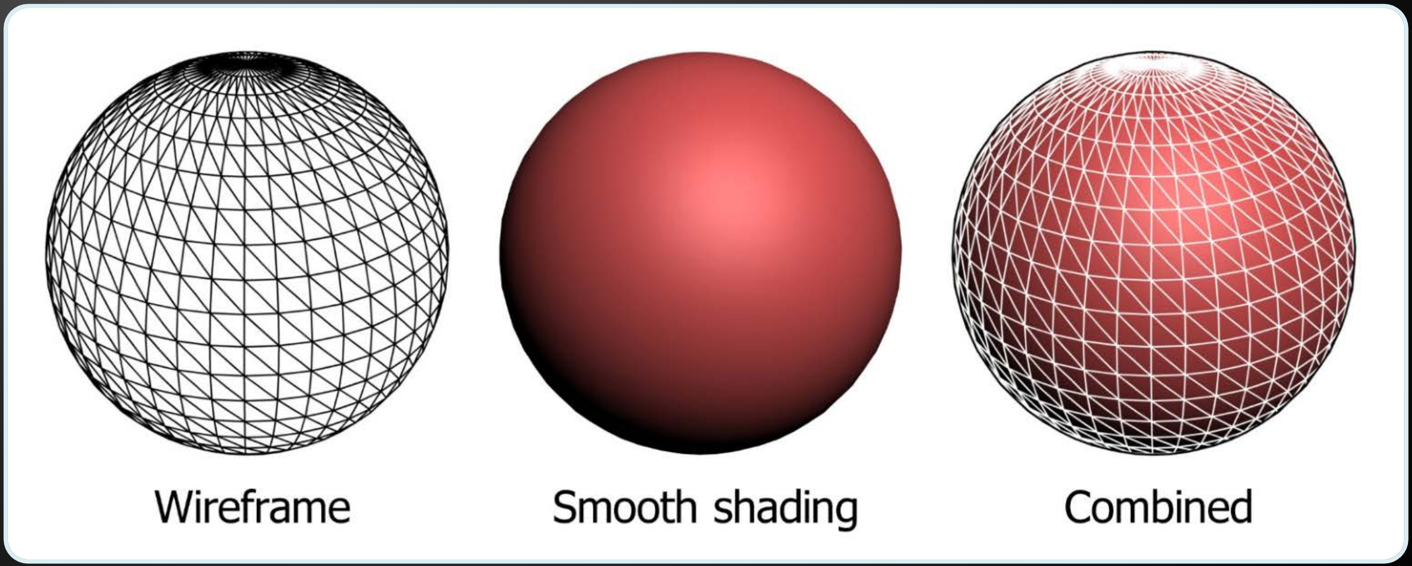
```
var view = {
students: [
  { name: "Pesho", marks: [5, 4, 3.5] },
  { name: "Gosho" },
  { name: "Mimeto", marks: [6.5, 5.3, 3.5] }
]
};

var template =
  "{{#students}} {{name}} has " +
    "{{#marks}} {{.}} {{/marks}}" +
    "{{^marks}} no marks {{/marks}}\n" +
  "{{/students}}";
```

# Inverted Sections

Live Demo

# Render Functions



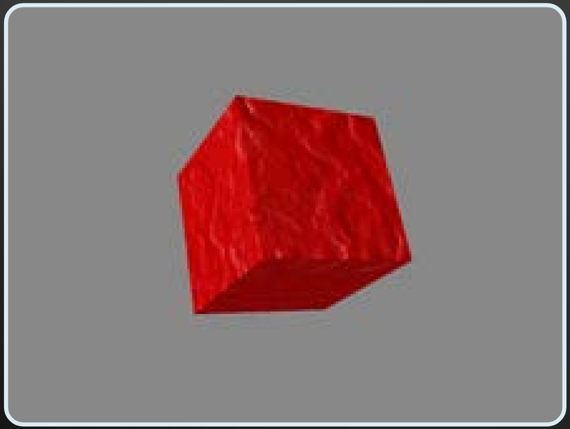
# Render Functions

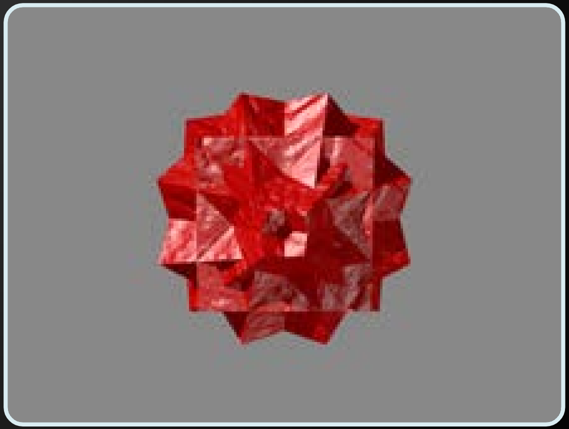
- Mustache.js can use render functions, if they are defined in the object
  - Define a render function to control the rendering of your object

```
var person = {
  fname: "Doncho", lname: "Minkov",
  fullname: function () {
    return this.fname + " " + this.lname; },
  render: function () {
    return function (text, render) {
      return "<strong>" + render(text) + "</strong>";
    }
  }
};
var template = "{{#render}} {{fullname}} {{/render}}";
var output = Mustache.render(template, donchoPerson);
```

# Render Functions

## Live Demo





# HTML Templates

# HTML Templates

- Mustache.js can render objects into HTML code
  - As in KendoUI, jQueryTemplates, AngularJS, etc.
- Create a script tag with an invalid type value
  - Put the template inside

```
<script type="text/x-must-tmpl" id="person-template">
  <strong>{{fullname}}</strong>
    <ul>
      {{#marks}}
        <li>{{subject}}: <strong>{{score}}</strong></li>
      {{/marks}}
      {{^marks}}
        <li>This student has no marks, yet</li>
      {{/marks}}
    </ul>
</script>
```

# HTML Templates (2)

- Implement a render function that takes a template

```
function render(items, template) {
  var list = document.createElement("ul");
  for (var i = 0; i < items.length; i++) {
    var listItem = document.createElement("li");
    var item = this.itemsSource[i];
    listItem.innerHTML = template(item);
    list.appendChild(listItem);
  }
  return list.outerHTML;
}
```

# HTML Templates (3)

- Using compiled templates, pass an item collection and the compiled template

```
var templateHTML =
   document.getElementById("person-template").innerHTML;
var template = Mustache.compile(templateHTML);

var people = […]

render(people, template);
```

# HTML Templates

Live Demo

# Questions?

# Homework

1. Change the ListView from the last demo to use TABLE instead of UL
   - The TableView gets number of rows and columns in its constructor
   - The TableView renders the elements in table cells
     - Each element gets only a single cell

# Homework (2)

- Create a web services that returns a list of students
  - Students have first and last name, grade, age and list of marks
  - Marks have subject and score

2. Create a Master-detail for the students and their marks
   - Create a template for rendering students
   - Create a template for rendering marks
   - When student is clicked, their marks must be rendered in another HTML element