Non classroom activity
Content representation languages

In this activity we want to build an HTML document and an XML with some code
Requisites: a web browser like Firefox or Chrome.

HTML Document

We start from an HTML document with a paragraph and an image. If you place it in the file called test.html and open it with your browser you will see the effect.

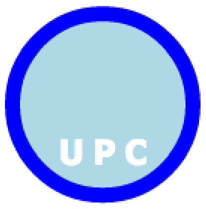
```
<!DOCTYPE html>
<html>
<body>
<p>Logo UPC</p>
<img src="upc.png"/>
</body>
</html>
```

The document has an HTML element of type 'p' that contains the text 'Logo UPC' and the element 'img' with the attribute src and value 'upc.png'. The termination '/>' is an abbreviated form of <img src = "upc.png"> </ img>. If you do not have an upc.png file will appear in your browser a mark where the image would go.

XML Document

Now we are going to replace the image 'img' of type PNG, by an image in vector format (Scalable Vector Graphics or SVG). SVG allows you to represent graphics in vector format encoded in XML. For example the code and its representation in your browser on the right would be:

```
<svg xmlns="http://www.w3.org/2000/svg" width="150" height="150">
<circle id="circleId" cx="75" cy="75" r="60" stroke="blue" stroke-width="10" fill="lightblue"></circle>
<text id="label_id" x="46" y="115" fill="white" font-size="22" font-weight="bold"
  font-family="Tahoma">U P C</text>
</svg>
```



It is an XML document that follows the specification (namespace) xmlns = "http://www.w3.org/2000/svg" and occupies an area of 150x150 pixels with:
- a circle centered on (75,75) and 60 pixels of radius, border of 10 pixels in blue and filled with light blue.
- a text with a series of attributes that puts UPC.
- and the document ends svg.

In fact, the official logo of the UPC is defined in SVG as follows:

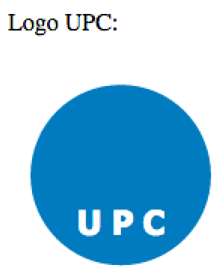
```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE svg PUBLIC "-//W3C//DTD SVG 1.1//EN" "http://www.w3.org/Graphics/SVG/1.1/DTD/svg11.dtd">
<svg version="1.1" id="Capa_1" xmlns="http://www.w3.org/2000/svg" x="0px" y="0px" width="150px" height="150px"
viewBox="0 0 150 150" enable-background="new 0 0 150 150">
	<circle fill="#007BC0" cx="75" cy="75" r="60"/>
	<path fill="#FFFFFF" d="M55.816,112.92c0.795-0.01,1.313-0.17,1.925-0.678c0.886-0.734,0.998-1.626,1.017-
2.77V97.455h5.764V109.5	c0,2.104-0.075,4.697-2.84,6.65c-1.803,1.271-3.939,1.537-5.868,1.537c-1.931,0-4.067-0.267-
5.869-1.537	c-2.766-1.953-2.84-4.549-2.84-6.65V97.455h5.763v12.02c0.018,1.144,0.131,2.033,1.018,2.771
C54.498,112.754,55.021,112.91,55.816,112.92"/>
	<path fill="#FFFFFF" d="M74.183,106.424v-4.645h2.229c1.605,0,2.078,1.045,2.117,2.248c0.041,1.231-0.57,2.396-
2.059,2.396H74.183	M69.096,117.266h5.027v-6.484l1.666,0.029c3.211,0,4.461-0.297,5.563-1.07c1.664-1.161,2.588-
3.122,2.588-5.473	c0-4.371-2.498-6.813-6.545-6.813h-8.296v19.811H69.096z"/>
	<path fill="#FFFFFF" d="M102.039,116.547c-1.887,0.928-2.971,1.192-4.791,1.192c-5.979,0-10.201-4.253-10.201-
10.291	c0-5.978,4.313-10.409,10.143-10.409c1.795,0,2.955,0.291,4.851,1.225v5.186c-1.964-1.646-2.738-1.53-4.285-
1.53	c-3.004,0-5.086,2.291-5.086,5.531c0,3.271,2.11,5.53,5.115,5.53c1.624,0,2.438-0.213,4.256-
1.565L102.039,116.547"/>
```

```
	<path fill="#FFFFFF" d="M45.86,40.502c0-4.73,3.834-8.565,8.563-8.565c4.73,0,8.564,3.834,8.564,8.565   c0,4.729-3.834,8.563-8.563,8.563C49.693,49.065,45.86,45.231,45.86,40.502"/>
	<path fill="#FFFFFF" d="M45.86,60.799c0-4.729,3.834-8.563,8.565-8.563c4.729,0,8.563,3.834,8.563,8.563 c0,4.73-3.834,8.565-8.563,8.565C49.693,69.364,45.86,65.529,45.86,60.799"/>
	<path fill="#FFFFFF" d="M45.86,81.633c0-4.729,3.834-8.563,8.565-8.563c4.729,0,8.563,3.834,8.563,8.563 c0,4.73-3.834,8.565-8.563,8.565C49.693,90.198,45.86,86.363,45.86,81.633"/>
	<path fill="#FFFFFF" d="M66.674,40.502c0-4.73,3.834-8.565,8.564-8.565c4.731,0,8.565,3.834,8.565,8.565 c0,4.729-3.834,8.563-8.564,8.563C70.51,49.065,66.676,45.231,66.674,40.502"/>
	<path fill="#FFFFFF" d="M66.676,60.799c0-4.729,3.834-8.563,8.564-8.563c4.729,0,8.563,3.834,8.563,8.563 c0,4.73-3.834,8.565-8.563,8.565C70.51,69.364,66.676,65.529,66.676,60.799"/>
	<path fill="#FFFFFF" d="M66.676,81.633c0-4.729,3.834-8.563,8.564-8.563c4.729,0,8.563,3.834,8.563,8.563 c0,4.73-3.834,8.565-8.563,8.565C70.51,90.198,66.676,86.363,66.676,81.633"/>
	<path fill="#FFFFFF" d="M87.01,40.502c0-4.73,3.834-8.565,8.563-8.565c4.729,0,8.564,3.834,8.564,8.565 c0,4.729-3.834,8.563-8.564,8.563C90.844,49.065,87.01,45.231,87.01,40.502"/>
	<path fill="#FFFFFF" d="M87.01,60.799c0-4.729,3.834-8.563,8.563-8.563c4.729,0,8.564,3.834,8.564,8.563 c0,4.73-3.834,8.565-8.564,8.565C90.844,69.364,87.01,65.529,87.01,60.799"/>
	<path fill="#FFFFFF" d="M87.01,81.633c0-4.729,3.834-8.563,8.563-8.563c4.729,0,8.564,3.834,8.564,8.563 c0,4.73-3.834,8.565-8.564,8.565C90.844,90.198,87.01,86.363,87.01,81.633"/>
</svg>
```

The former code tells us that it is an XML document, encoded in UTF-8. The type of document is SVG according to specific specifications that details (version 1.1 of SVG).
This document contains an XML tree with a <svg> root element containing the circle of the logo of the color "# 007BC0" centered on the coordinates 75,75 and radius 60.
Following are several white " #FFFFFF " objects as paths with the letters 'U', 'P', 'C' and each of the 9 circles.
Each <path> is expressed as a series of commands to draw. For example the first one starts with 'M' (moveto) at the coordinates 55.816.112.92 followed by 'c' a curve ... The format is complicated so it is recommended to use an SVG editor such as InkScape or LibreOffice to view and edit each element of the picture.

Returning to a simple document, we will try to draw the UPC logo on an HTML page that contains an SVG document with its representation in your browser as shown on the right:

```
<!DOCTYPE html>
<html>
<body>
<p>Logo UPC:</p>
<svg id="drawing" width="150" height="150">
<circle fill="#007BC0" cx="75" cy="75" r="60"/>
<text id="label_id" x="46" y="115" fill="white" font-size="22" font-weight="bold"
  font-family="Tahoma">U P C</text>
</svg>
</body>
```



To complete the UPC logo you have to add several circles.
To do this we create the createCircle function in Javascript that returns an SVG circle element with the correct name space and the attributes of position, radius and color:

```
function createCircle(x, y, r) {
  var e = document.createElementNS("http://www.w3.org/2000/svg", 'circle');
  e.setAttribute("cx", x);
  e.setAttribute("cy", y);
  e.setAttribute("r", r);
  e.setAttribute("fill", "white");
  return e;
}
```

When loading the HTML page just be warned to run the init () function:

```
<body onload="init();">
```

which is as follows:

```
function init() {
  var svg = document.getElementById('drawing');

  for (var x=0; x < 3; x++) {
    for (var y=0; y < 3; y++) {
      var c = createCircle(54 + x*21, 46 + y*20, 8);
      svg.appendChild(c);
    }
  }
  var text = new XMLSerializer().serializeToString(svg);
  console.log(text);
}
```

The function searches the HTML document for the element with id = "drawing", then traverses the coordinates of each circle and adds it to the drawing (the root of the SVG). It also calls a function to convert the resulting SVG to text and shows it through the console (you can see it in Firefox: *Tools→Web Developer→Debugger, o en Chrome: View→Developer→Javascript console)*

The result is the following:

```
<svg xmlns="http://www.w3.org/2000/svg" id="drawing" width="150" height="150">
  <circle fill="#007BC0" cx="75" cy="75" r="60"/>
  <text id="label_id" x="46" y="115" fill="white" font-size="22" font-weight="bold" font-family="Tahoma">U P C</text>
  <circle cx="54" cy="46" r="8" fill="white"/>
  <circle cx="54" cy="66" r="8" fill="white"/>
  <circle cx="54" cy="86" r="8" fill="white"/>
  <circle cx="75" cy="46" r="8" fill="white"/>
  <circle cx="75" cy="66" r="8" fill="white"/>
  <circle cx="75" cy="86" r="8" fill="white"/>
  <circle cx="96" cy="46" r="8" fill="white"/>
  <circle cx="96" cy="66" r="8" fill="white"/>
  <circle cx="96" cy="86" r="8" fill="white"/>
</svg>
```

And the final result is:

```
<!DOCTYPE html>
<head>
<script>
function init() {
  var svg = document.getElementById('drawing');
  for (var x=0; x < 3; x++) {
    for (var y=0; y < 3; y++) {
      var c = createCircle(54 + x*21, 46 + y*20, 8);
      svg.appendChild(c);
    }
  }
  var text = new XMLSerializer().serializeToString(svg);
  console.log(text);
}

function createCircle(x, y, r) {
   var e = document.createElementNS("http://www.w3.org/2000/svg", 'circle');
   e.setAttribute("cx", x);
   e.setAttribute("cy", y);
   e.setAttribute("r", r);
   e.setAttribute("fill", "white");
   return e;
}
</script>
</head>
<body onload="init();">
<p>Logo UPC:</p>
<svg id="drawing" width="150" height="150">
<circle fill="#007BC0" cx="75" cy="75" r="60"/>
<text id="label_id" x="46" y="115" fill="white" font-size="22" font-weight="bold" font-family="Tahoma">U P C</text>
</svg>
</body>
```

Evaluation

Nothing must be delivered. The session will be evaluated the same day of the final lab exam, by means of a quiz. In the following you have some examples of the type of questions that you will find in the quiz.

Given the following XML document:

```
<svg xmlns="http://www.w3.org/2000/svg" id="drawing" width="150" height="150">
  <circle fill="#007BC0" cx="75" cy="75" r="60"/>
  <text id="label_id" x="46" y="115" fill="white" font-size="22" font-weight="bold" font-family="Tahoma">U P C</text>
  <circle cx="54" cy="46" r="8" fill="white"/>
```

Say which of the following statements are true:
1) It is well formed (false)
2)The element </svg> is missing in the end (true)
3) The element at the beginning <svg> is not needed (false)
4) The attribute xmlns allows verifying the validity with respect to an SVG specification (true)

References:

- SVG: https://www.w3.org/TR/SVG/ (Version 1.1, 2011)
- SVG Editors: https://en.wikipedia.org/wiki/Comparison_of_vector_graphics_editors
- Javascript basics: https://developer.mozilla.org/en-US/docs/Learn/Getting_started_with_the_web/JavaScript_basics
- Javascript: https://en.wikipedia.org/wiki/JavaScript
- HTML: https://www.w3schools.com/html/html_intro.asp