WEB Services

Niyazi ARI *Prof. Dr. sch. techn. ETH ,* Moussa Mahamat Boukar, *Ass. Professor*

***Abstract*—WEB Services convert the applications into a WEB application, which can publish its function or message to the rest of world. The basic WEB Services platform is XML + HTTP The types of WEB Services have been explained. WEB Services platform elements are illustrated. The ScalabelVector Graphics (SVG) is explained in detail. The advantages of SVG are illustrated using practical examples, such as SVG-Ellipse, SVG-Line, SVG-Rect, ..**

***Keywords*—WEB Services, SOAP, WSDL, UDDI, XML, HTML**

I. WEB SERVICES INTRODUCTION

Web Services can convert your application into a Web-application, which can publish its function or message to the rest of the world. The basic Web Services platform is XML + HTTP.

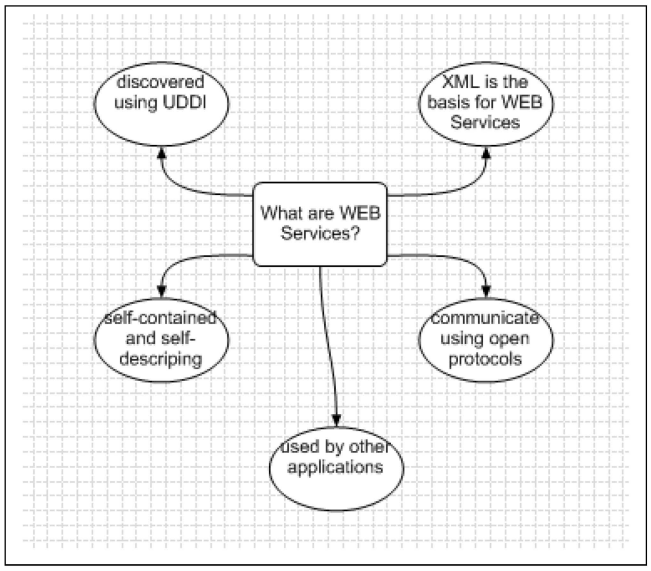

Fig. 1. Web Services

The Nagerian Turkish Nile University, Abuja, Nigeria
Manuscript received April 19, 2014

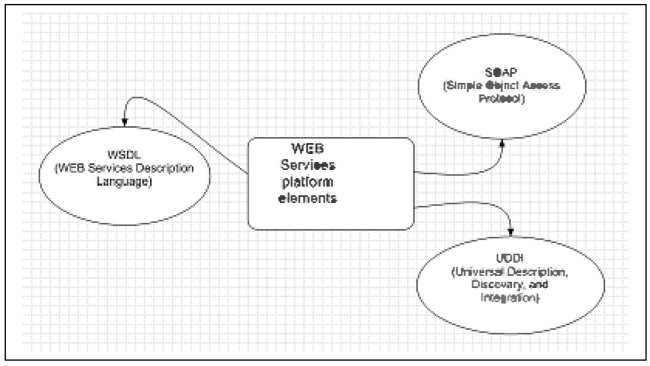

Fig. 2. Web Services

A. Interoperability has Highest Priority

When all major platforms could access the Web using Web browsers, different platforms could interact. For these platforms to work together, Web-applications were developed. Web-applications are simple applications that run on the web. These are built around the Web browser standards and can be used by any browser on any platform.

Web Services take Web-applications to the Next Level By using Web services, an application can publish its function or message to the rest of the world. Web services use XML to code and to decode data, and SOAP to transport it (using open protocols). With Web services, an accounting department's Win 2k server's billing system can connect with anr IT supplier's UNIX server.

B. Web Services have Two Types of Uses

Reusable application-components

There are things applications need very often. So why make these over and over again?

Web services can offer application-components like: currency conversion, weather reports, or even language translation as services.

Connect existing software:

Web services can help to solve the interoperability problem by giving different applications a way to link their data.

With Web services data can be exchanged data between different applications and different platforms.

II. WEB SERVICES PLATFORM ELEMENTS

Web Services have three basic platform elements: SOAP, WSDL and UDDI.

SOAP is an XML-based protocol to let applications exchange information over HTTP.

978-1-4799-4106-3/14/$31.00 © 2014 IEEE

Or more simple: SOAP is a protocol for accessing a Web Service.

WSDL is an XML-based language for locating and describing Web services.

UDDI is a directory service where companies can register and search for Web services.

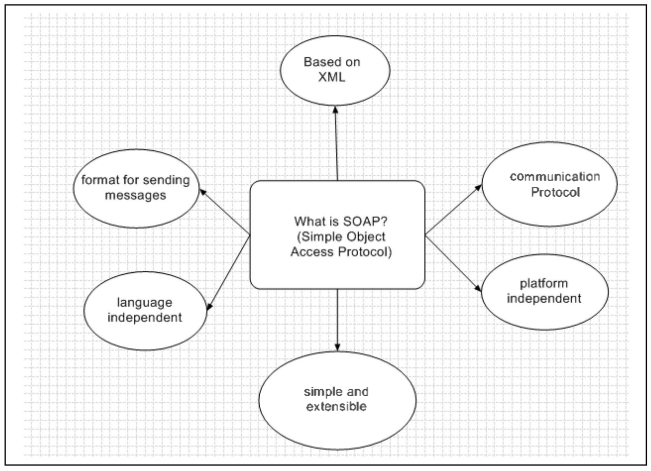

Fig. 3. SOAP

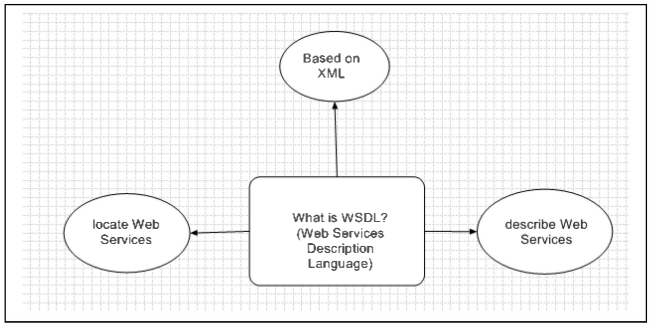

Fig. 4. WSDL

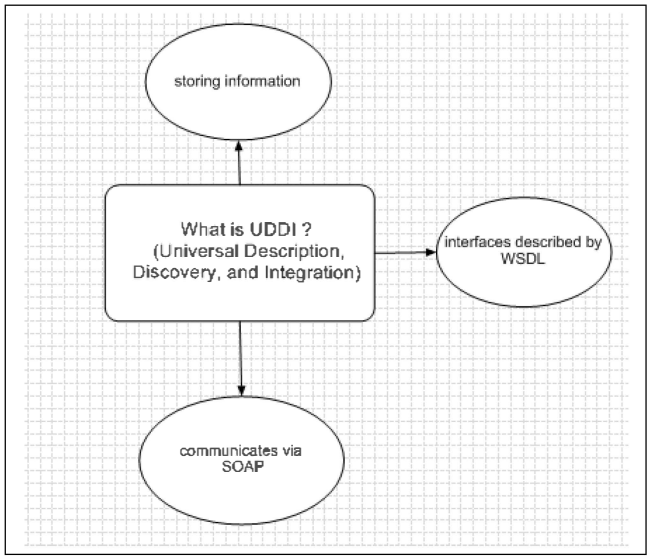

Fig. 5. UDDI

III. SVG Scalable Vector Graphics

SVG defines graphics in XML format.
SVG is a language for describing 2D-graphics and graphical applications in XML.

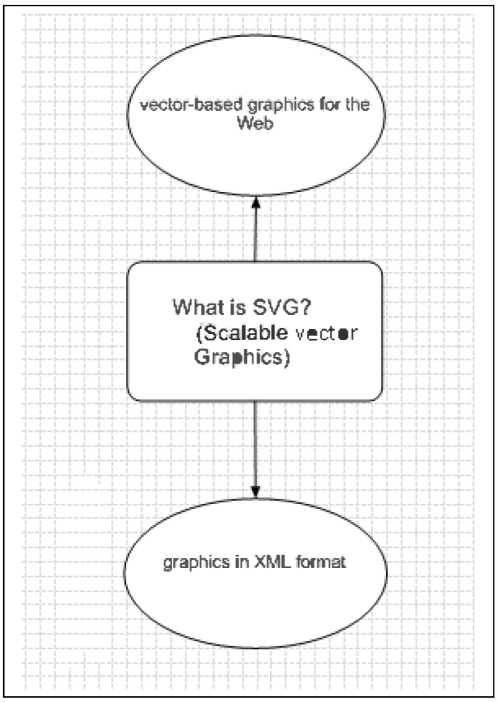

Fig. 6. SVG Scalable Vector Graphics

SVG Advantages

Sun Microsystems, Adobe, Apple, IBM, and Kodak are some of the organizations that have been involved in defining SVG.

Advantages of using SVG over other image formats (like JPEG and GIF) are:
SVG files can be read and modified by a large range of tools (e.g. notepad)
SVG files are smaller and more compressible than JPEG and GIF images
SVG images are scalable
SVG images can be printed with high quality at any resolution
SVG images are zoomable (and the image can be zoomed without degradation)
Text in SVG is selectable and searchable (excellent for making maps)
SVG works with Java technology
SVG is an open standard
SVG files are pure XML
The main competitor to SVG is Flash.
The biggest advantage SVG has over Flash is the compliance with other standards (e.g. XSL and the DOM). Flash relies on proprietary technology that is not open source.

SVG Shapes
SVG has some predefined shape elements that can be used and manipulated by developers.

SVG has some predefined shape elements that can be used and manipulated by developers:

-Rectangle <rect>
-Circle <circle>
-Ellipse <ellipse>
-Line <line>
-Polyline <polyline>
-Polygon <polygon>
-Path <path>

IV. SVG <ELLIPSE >

The <ellipse >tag is used to create an ellipse.

The <ellipse >Tag
The <ellipse >tag is used to create an ellipse. An ellipse has an x and a y radius that differs from each other. Copy the following code into Notepad and save the file as "ellipse1.svg". Place the file in Web directory:

```
<?xml version="1.0" standalone="no"?>
<!DOCTYPE svg PUBLIC "-//W3C//DTD SVG 1.1//
EN"
"http://www.w3.org/Graphics/SVG/1.1/DTD/
svg11.dtd">

<svg width="100%" height="100%" version=
"1.1"
xmlns="http://www.w3.org/2000/svg">

<ellipse cx="300" cy="150" rx="200" ry="100"
style="fill:rgb(200,100,50);
stroke:rgb(0,0,100);stroke-width:2"/>
</svg>
```

Code explanation:
- The cx attribute defines the x coordinate of the center of the ellipse
- The cy attribute defines the y coordinate of the center of the ellipse
- The rx attribute defines the horizontal radius
- The ry attribute defines the vertical radius

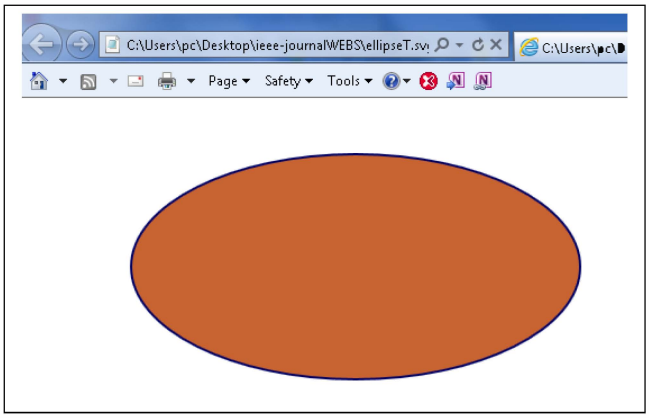

Fig. 7. SOAP

View example

The following example creates two ellipses :

```
<?xml version="1.0" standalone="no"?>
<!DOCTYPE svg PUBLIC "-//W3C//DTD SVG 1.1//
EN"
"http://www.w3.org/Graphics/SVG/1.1/DTD/
svg11.dtd">
<svg width="100%" height="100%" version=
"1.1"
xmlns="http://www.w3.org/2000/svg">
<ellipse cx="240" cy="100" rx="220" ry="30"
style="fill:yellow"/>
<ellipse cx="220" cy="100" rx="190" ry="20"
style="fill:red"/>
</svg>
```

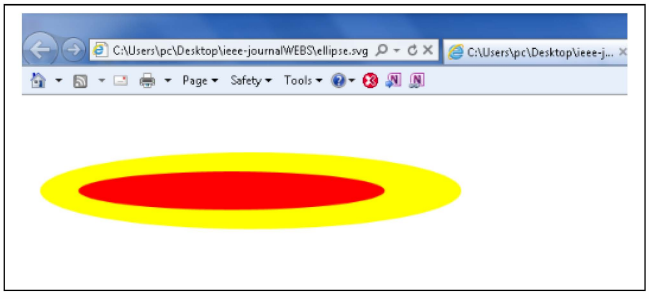

Fig. 8. SOAP

V. SVG <LINE >

The <line >tag is used to create a line.

The <line >Tag

The <line >tag is used to create a line.
Copy the following code into Notepad and save the file as ”line1.svg”. Place the file in your Web directory:

```
<?xml version="1.0" standalone="no"?>
<!DOCTYPE svg PUBLIC "-//W3C//DTD SVG 1.1//
EN"
"http://www.w3.org/Graphics/SVG/1.1/DTD/
svg11.dtd">
<svg width="100%" height="100%" version=
"1.1"
xmlns="http://www.w3.org/2000/svg">
<line x1="10" y1="10" x2="200" y2="200"
style="stroke:rgb(99,99,99);stroke-width:2"/>
</svg>
```

Code explanation:

- The x1 attribute defines the start of the line on the x-axis
- The y1 attribute defines the start of the line on the y-axis
- The x2 attribute defines the end of the line on the x-axis
- The y2 attribute defines the end of the line on the y-axis

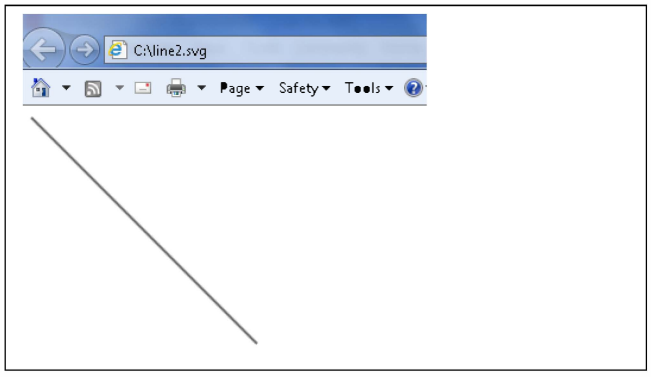

Fig. 9. SOAP

VI. SVG <POLYGON >

The <polygon >tag is used to create a graphic that contains at least three sides.

The <polygon >Tag

The <polygon >tag is used to create a graphic that contains at least three sides.
Copy the following code into Notepad and save the file as ”polygon1.svg”. Place the file in your Web directory:

```
<?xml version="1.0" standalone="no"?>
<!DOCTYPE svg PUBLIC "-//W3C//DTD SVG 1.1
//EN"
"http://www.w3.org/Graphics/SVG/1.1/DTD/
svg11.dtd">
<svg width="100%" height="100%" version=
"1.1"
xmlns="http://www.w3.org/2000/svg">
<polyline points="0,0 0,30 30,30 30,40 40,
50 50,60"
style="fill:white;stroke:red;stroke-width:2"/>
</svg>
```

Code explanation:

- The points attribute defines the x and y coordinates for each corner of the polygon

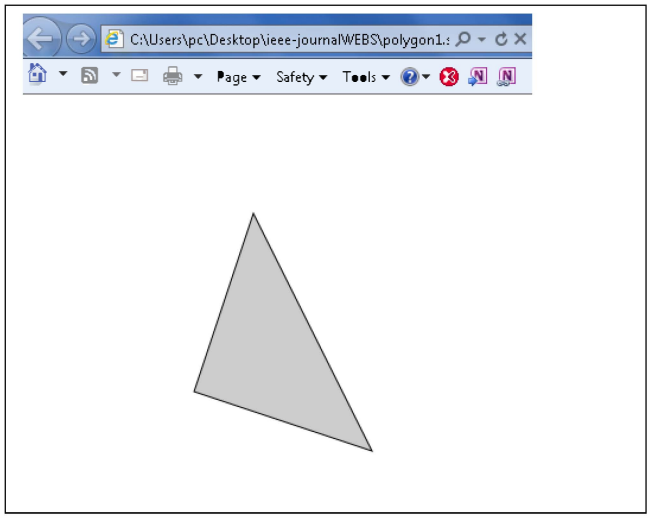

Fig. 10. SOAP

VII. SVG <POLYLINE >

The <polyline >tag is used to create any shape that consists of only straight lines.

The <polyline >Tag

The <polyline >tag is used to create any shape that consists of only straight lines.
Copy the following code into Notepad and save the file as ”polyline1.svg”. Place the file in your Web directory:

```
<?xml version="1.0" standalone="no"?>
<!DOCTYPE svg PUBLIC "-//W3C//DTD SVG 1.1
//EN"
"http://www.w3.org/Graphics/SVG/1.1/DTD/
svg11.dtd">
<svg width="100%" height="100%" version=
"1.1"
xmlns="http://www.w3.org/2000/svg">
<polyline points="0,100 100,200 200,100 100,
300 450,250 550,600"
style="fill:white;stroke:red;stroke-width:2"/>
</svg>
```

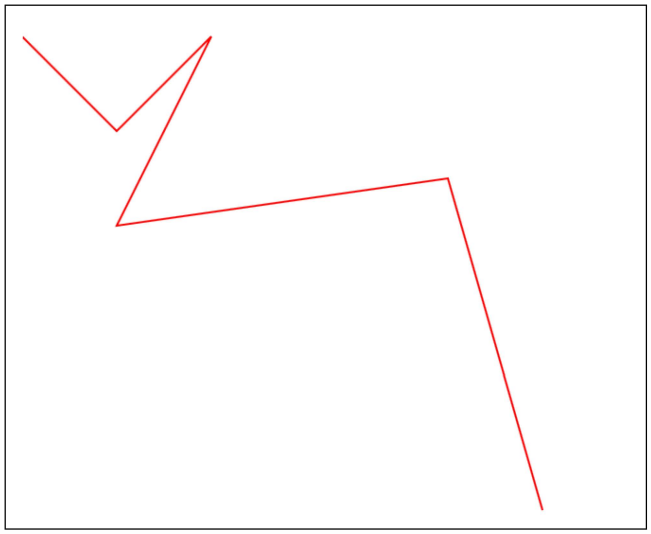

Fig. 11. SOAP

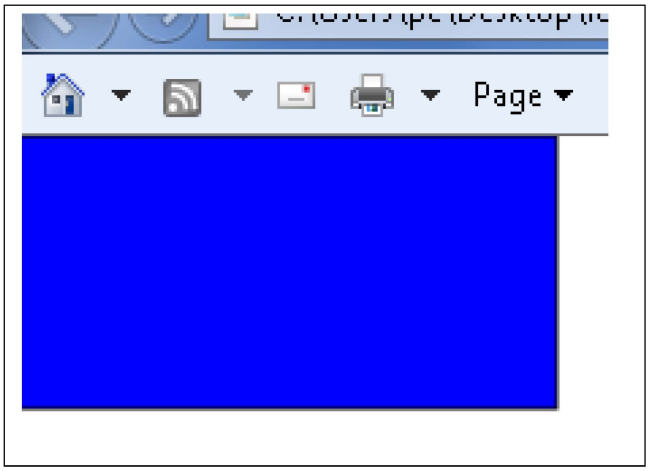

Fig. 12. Rect1

VIII. SVG <RECT >

The <rect >Tag

The <rect >tag is used to create a rectangle and variations of a rectangle shape. To understand how this works, copy the following code into Notepad and save the file as ”rect1.svg”. Place the file in your Web directory:

```
<?xml version="1.0" standalone="no"?>
<!DOCTYPE svg PUBLIC "-//W3C//DTD SVG 1.1//
EN"
"http://www.w3.org/Graphics/SVG/1.1/DTD/
svg11.dtd">

<svg width="100%" height="100%" version=
"1.1"
xmlns="http://www.w3.org/2000/svg">

<rect width="200" height="100"
style="fill:rgb(0,0,255);stroke-width:1;
stroke:rgb(0,0,0)"/>

</svg>
```

Code explanation:

- The width and height attributes of the rect element define the height and the width of the rectangle
- The style attribute is used to define CSS properties
- The CSS fill property defines the fill color of the rectangle (either an rgb value, a color name, or a hexadecimal value)
- The CSS stroke-width property defines the width of the border of the rectangle
- The CSS stroke property defines the color of the border of the rectangle

View example

Let’s look at another example that contains some new attributes:

```
<?xml version="1.0" standalone="no"?>
<!DOCTYPE svg PUBLIC "-//W3C//DTD SVG 1.1/
/EN"
"http://www.w3.org/Graphics/SVG/1.1/DTD/
svg11.dtd">

<svg width="100%" height="100%" version=
"1.1"
xmlns="http://www.w3.org/2000/svg">

<rect x="10" y="10" width="200" height=
"250"
style="fill:blue;stroke:pink;stroke-width:5;
fill-opacity:0.1;stroke-opacity:0.9"/>

</svg>
```

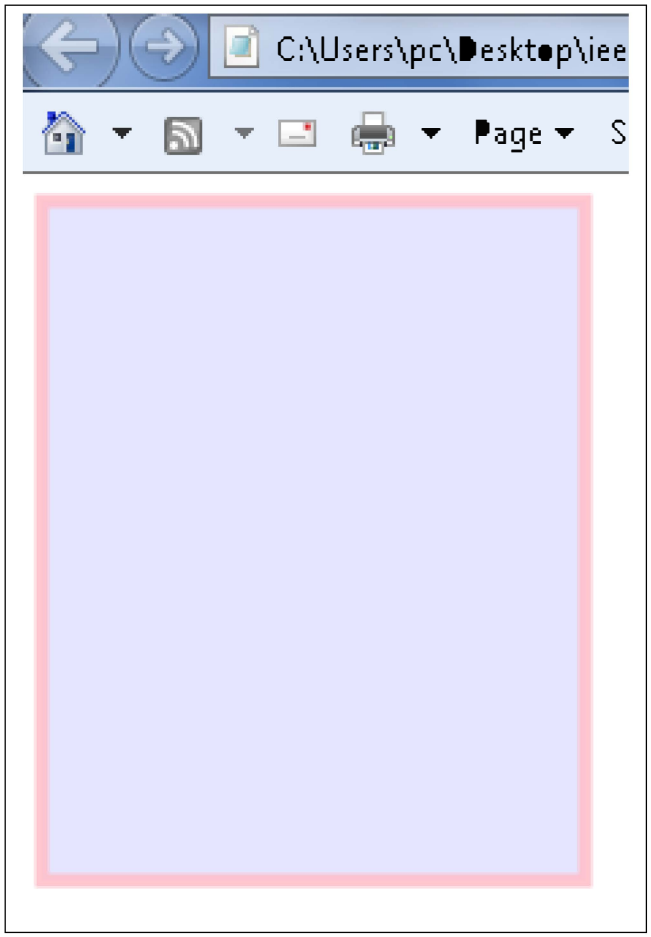

Fig. 13. Rect2

Code explanation:
- The x attribute defines the left position of the rectangle (e.g. x=”0” places the rectangle 0 pixels from the left of the browser window)
- The y attribute defines the top position of the rectangle (e.g. y=”0” places the rectangle 0 pixels from the top of the browser window)
- The CSS fill-opacity property defines the opacity of the fill color (legal range: 0 to 1)
- The CSS stroke-opacity property defines the opacity of the stroke color (legal range: 0 to 1)

View example

Define the opacity for the whole element:

```
<?xml version="1.0" standalone="no"?>
<!DOCTYPE svg PUBLIC "-//W3C//DTD SVG 1.1//
EN"
"http://www.w3.org/Graphics/SVG/1.1/DTD/
svg11.dtd">

<svg width="100%" height="100%" version=
"1.1"
xmlns="http://www.w3.org/2000/svg">

<rect x="10" y="10" width="200" height=
"200"
style="fill:blue;stroke:pink;stroke-width:5;
opacity:0.9"/>

</svg>
```

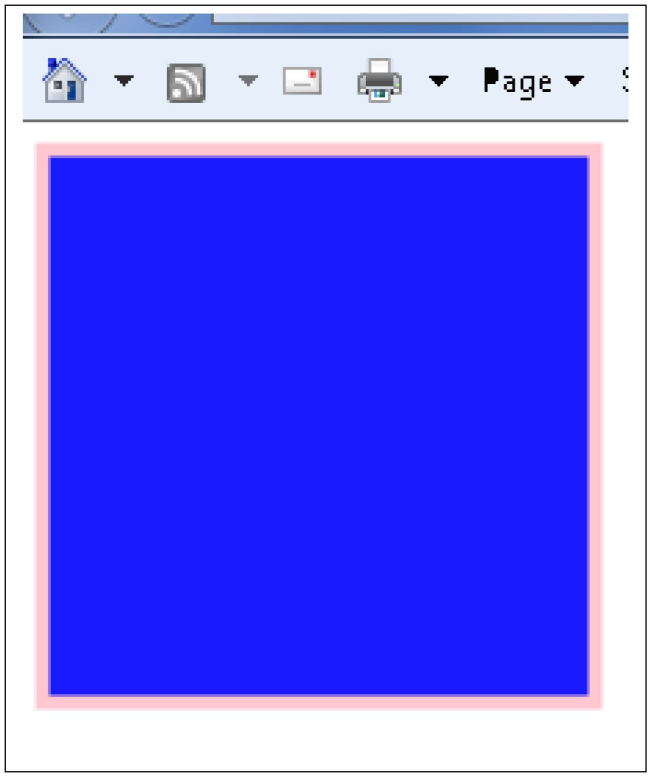

Fig. 14. Rect3

Code explanation:
- The CSS opacity property defines the opacity value for the whole element (legal range: 0 to 1)

View example

```
Last example, create a rectangle with rounded
 corners:\\
\begin{verbatim}
<?xml version="1.0" standalone="no"?>
<!DOCTYPE svg PUBLIC "-//W3C//DTD SVG 1.1//
EN"
"http://www.w3.org/Graphics/SVG/1.1/DTD/
svg11.dtd">

<svg width="100%" height="100%" version=
"1.1"
xmlns="http://www.w3.org/2000/svg">
```

```
<rect x="30" y="30" rx="30" ry="30" width=
"250"
height="100" style="fill:red;stroke:black;
stroke-width:5;opacity:0.5"/>

</svg>
```

Code explanation:

- The rx and the ry attributes rounds the corners of the rectangle

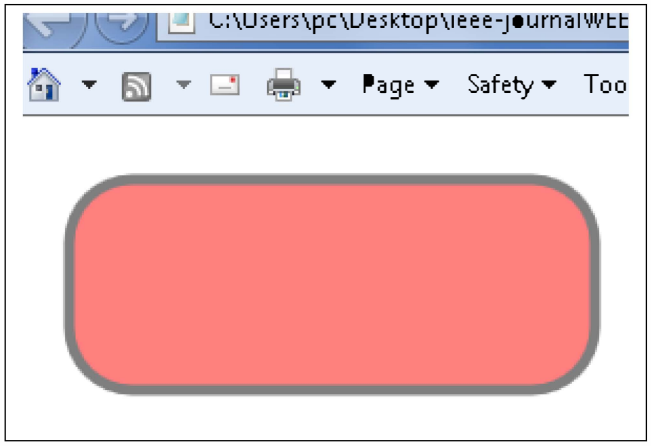

Fig. 15. Rect4

IX. Conclusion

WEB Services convert the applications into a WEB application, which can publish its function or message to the rest of world. The basic WEB Services platform is XML + HTTP The types of WEB Services have been explained. WEB Services platform elements are illustrated. The Scalabel Vector Graphics (SVG) is explained in detail. The advantages of SVG are illustrated using practical examples, such as SVG-Ellipse, SVG-Line, SVG-Rect, ..
The Knowledge of WEB Services can be used in different WEB Projects. It is possible in this domain to define some additional research projects.

References

[1] w3schools.com

[2] www.w3.org

[3] Niyazi ARI, WEB Services Lecture Notes, University of Technology,Zurich, Switzerland, 2009