CSIT6910 Independent Project

Spring 2013

Dynamic SVG Comic

Student: CHENG Xu

Supervisor: Prof. David Rossiter

1. Introduction

1.1 Background

a) SVG Introduction

SVG (Scalable Vector Graphics) **[1]** is a language for describing two-dimensional graphics in XML. SVG allows for three types of graphic objects: vector graphic shapes (e.g., paths consisting of straight lines and curves), images and text. Graphical objects can be grouped, styled, transformed and composited into previously rendered objects. The feature set includes nested transformations, clipping paths, alpha masks, filter effects and template objects.

b) Dynamic and Interactive feature

SVG drawings can be interactive and dynamic. We can define and trigger animation either declaratively (i.e., by embedding SVG animation elements in SVG content) or via scripting.

In use of the DOM (Document Object Model) **[2]** port to code, we can accomplish strong dynamics and interaction including generate webpages with SVG image dynamically, response to operations like highlight, sound effect, animation effect. This features are especially contained in SVG, which GIF and JPEG doesn't have. SVG also defines abundant events, such as mouse events and keyboard events. Interaction can be implement in SVG file when making some script programing. The implementation of animation is more feasible. When insert some animation elements in SVG file, the animation effects can be achieved.

c) Film Introduction

WALL-E is one of the most famous science-fiction comedy film produced by Pixar Animation Studios. Not only the brilliant story plot, the excellent animation effects is very attractive, but also the environment protection theme arouses deep thoughts. The reason why I choose this film as the theme of my SVG comic is that this film is well-known by almost all contemporary people. The comic can by well-understood while not displaying all the plots. Another reason why this film is suitable for animation is that its characters are simple and vivid. I can cut the figure easily without caring too much about the details.

1.2 Objective

This independent project aims at Creating SVG Comic on a theme of the movie Wall-E. Making use of SVG's strong advantages in mapping and animation, this independent project is meaningful in exploring this less popular area. Because the SVG file is tiny enough and easy to be uploaded and downloaded, the display on webpage can be fast and rapidly spread. Thus, the study of this project can be extended to many areas, like advertising and webpage design.

2. General Idea

There are 21 scenes named from 0.svg to 20.svg in this project. One HTML controls all the scenes by running them in a recursion way. The whole comic applies the SVG animation elements to generate animation effects. I use the movie Wall-E's screen cut as raw materials to create the background of scenes. The character's .png images are cut using the scissors selection tool of GIMP (figure 1). Every scene is written in notepad, and saved as num.svg file (figure 2), so that the main program can call them by their number. Originally I used Chrome browser to run and check the project, so all effects can be perfectly displayed in Chrome. As for SVG files can be theoretically run on all browsers, this SVG Comic can be displayed on Internet Explorer, Opera, and Firefox as well.

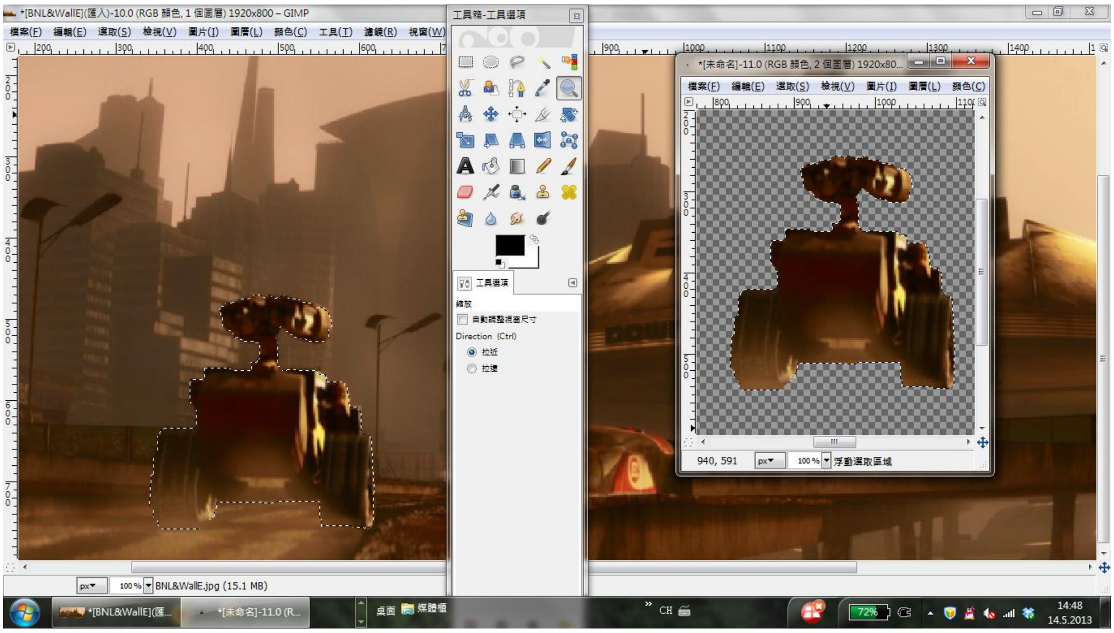

*[BNL&WallE](匯入)-10.0 (RGB 顏色, 1 個圖層) 1920x800 – GIMP
檔案(F) 編輯(E) 選取(S) 檢視(V) 圖片(I) 圖層(L) 顏色(C) 工具(T) 濾鏡(R) 視窗(W)
工具箱-工具選項
工具選項
縮放
自動調整視窗尺寸
Direction (Ctrl)
拉近
拉遠
*[未命名]-11.0 (RGB 顏色, 2 個圖層) 1920x80...
940, 591
px
100 %
浮動選取區域
BNL&WallE.jpg (15.1 MB)
*[BNL&WallE](匯...
*[未命名]-11.0 (R...
桌面
媒體櫃
CH
72%
14:48
14.5.2013

Figure 1. Separating character from scene in GIMP

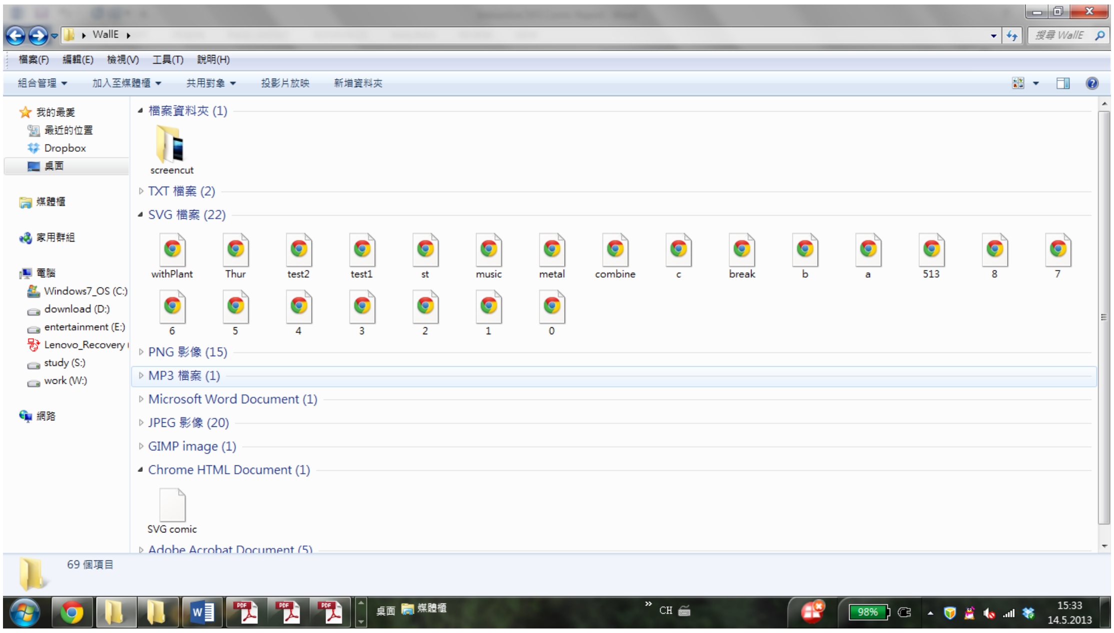
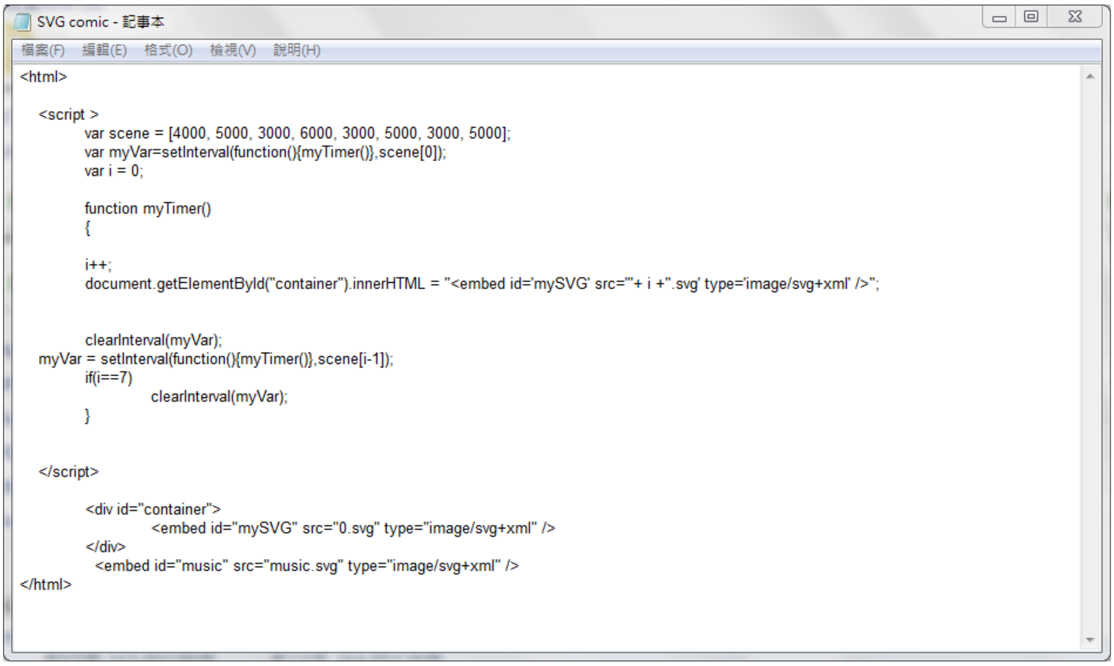


Figure 2. the files



```
<html>

    <script >
            var scene = [4000, 5000, 3000, 6000, 3000, 5000, 3000, 5000];
            var myVar=setInterval(function(){myTimer()},scene[0]);
            var i = 0;

            function myTimer()
            {

            i++;
            document.getElementById("container").innerHTML = "<embed id='mySVG' src='"+ i +".svg' type='image/svg+xml' />";


            clearInterval(myVar);
    myVar = setInterval(function(){myTimer()},scene[i-1]);
            if(i==7)
                        clearInterval(myVar);
            }


    </script>

            <div id="container">
                        <embed id="mySVG" src="0.svg" type="image/svg+xml" />
            </div>
              <embed id="music" src="music.svg" type="image/svg+xml" />
</html>
```

Figure 3. The main HTML file

3. Structure

3.1 Project Structure

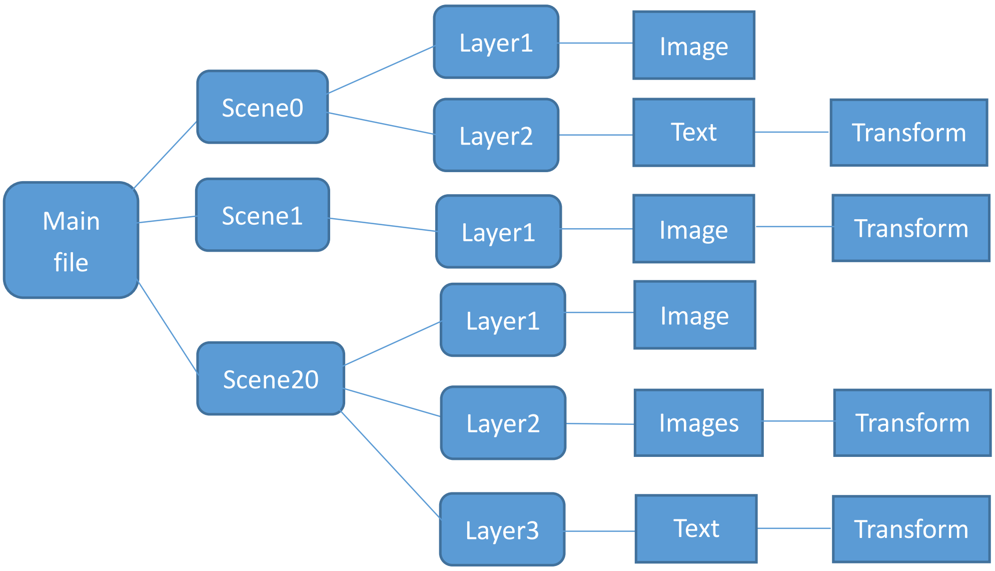


Figure 4. The structure of the whole program

In this project, I use 21 scenes to tell the story. The HTML file works as the main function. It runs the SVG files 0 to 20, at different time durations which are set in an array.

Every SVG file includes at least one layer inside the g element. The first layer is always the background cut from the original movie. The second and third layers are the image or text which can move around or interact with each other. In the SVG file, the image or text are written under the layers. The transformation sentence are always following the image or text.

3.2 SVG File Example Structure

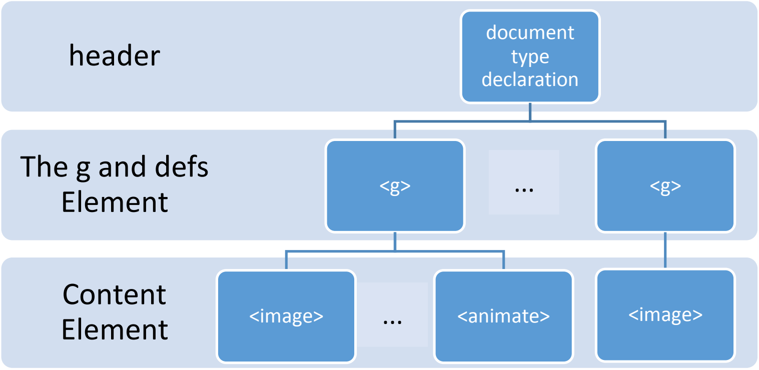
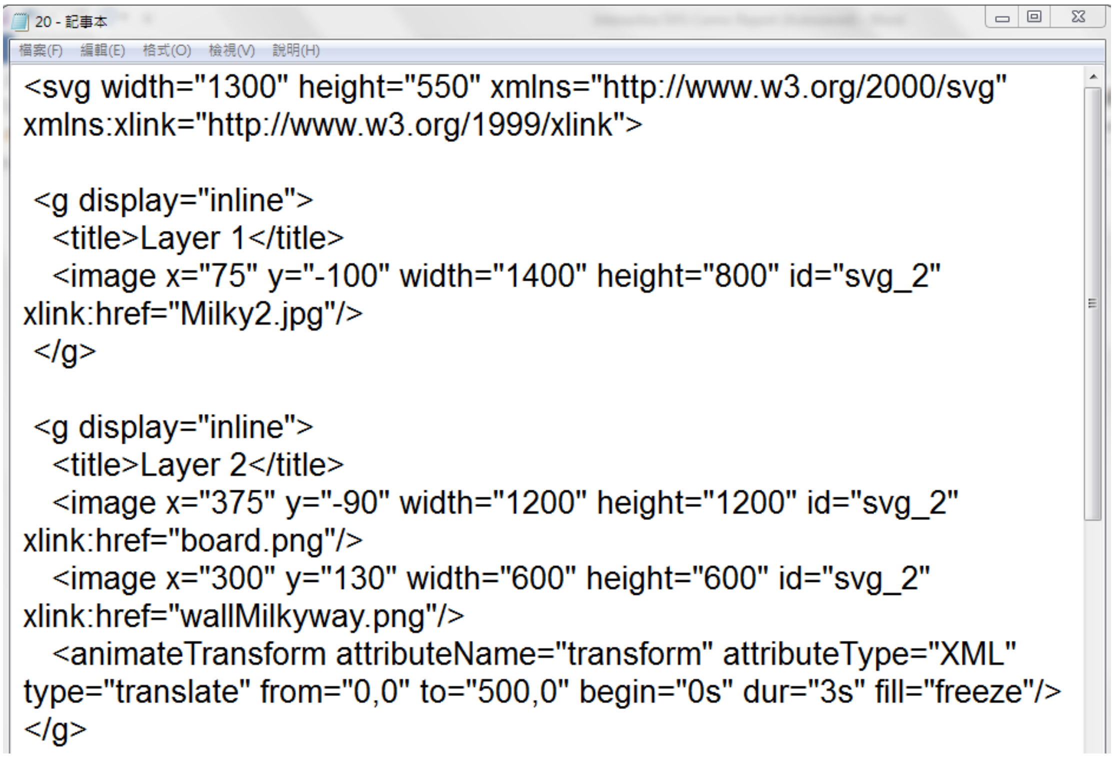


Figure 5. The structure of 20.SVG file

The <g> element gathers all of its child elements as a group, and often has an id attribute to give that group a unique name. Furthermore, each group may have its own <title> and<desc> to identify it for text-based XML applications or to aid in accessibility for visually-impaired users. In addition to the conceptual clarity that comes from the ability to group and document objects, the <g> element also provides notational convenience. Any styles you specify in the starting <g> tag will apply to all the child elements in the group.

An animation element is an element that can be used to animate the attribute or property value of another element. The following elements are animation elements: ‘animateColor’, ‘animateMotion’, ‘animateTransform’, ‘animate’ and ‘set’.



```
<svg width="1300" height="550" xmlns="http://www.w3.org/2000/svg"
xmlns:xlink="http://www.w3.org/1999/xlink">

 <g display="inline">
   <title>Layer 1</title>
   <image x="75" y="-100" width="1400" height="800" id="svg_2"
xlink:href="Milky2.jpg"/>
 </g>

 <g display="inline">
   <title>Layer 2</title>
   <image x="375" y="-90" width="1200" height="1200" id="svg_2"
xlink:href="board.png"/>
   <image x="300" y="130" width="600" height="600" id="svg_2"
xlink:href="wallMilkyway.png"/>
   <animateTransform attributeName="transform" attributeType="XML"
type="translate" from="0,0" to="500,0" begin="0s" dur="3s" fill="freeze"/>
</g>
```

Figure 6. The 20.SVG file

4. Technical Detail

4.1 SVG Animation method

SVG content can be animated using SVG's animation elements. The various elements can define transformation, motion paths, fade-in or fadeout effects, and objects that grow, shrink, spin or change color. The animation features of SVG are based on the World Wide Web Consortium’s Synchronized Multimedia Integration Language Level 3 (SMIL3) **[3]**. In this system, you specify the starting and ending values of the attribute, color, motion, or transformation that you with to animate; the time at which the animation should begin; and the duration of the animation **[4]**.

4.2 Practice

a) Use translate with the <animationTransform>:

Rotate:

```
<animateTransform attributeName="transform" attributeType="XML"
type="rotate" from="-15,364,318" to="15, 364, 318" begin="0"
dur="1.5s" repeatCount="2" fill="freeze"/>
```

This sentence means rotate the object at position (364,318), from 15 degree to -15 degree, and repeat this motion twice.

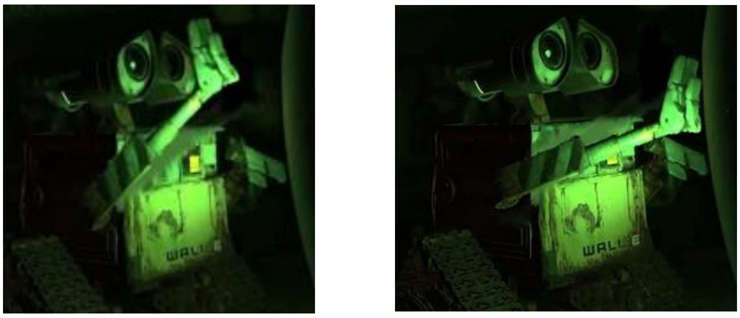


Figure 7. The rotation of Wall-E's arm

The combination of scale and translate:

```
<animateTransform attributeName="transform" attributeType="XML"
type="rotate" from="0 1280 560" to="28 1280 560" begin="1s"
dur="2s" fill="freeze" />
<animateTransform attributeName="transform" attributeType="XML"
type="translate" from="0,0" to="420,105" additive="sum" begin="1s"
dur="2s" fill="freeze" />
```

This two sentences consist the command 'additive="sum" ', which means combine the motion translate and scale together and apply to one object.

b) Use <animate Motion> to make figure move in a path:

```
<animateMotion path="M5,10 L5,-10, L5,30 L5,-10, L5,30 L10,-10,
L10,30 L10,-10, L10,30 L15,-10, L15,30 L15,-10, L15,30 L20,-10,
L20,30 L20,-10, L20,30 L25,-10, L25,30 L25,-10, L25,30 L30,-10,
L30,30" begin="0s" dur="3s" fill="freeze"/>
```

This sentence gives a motion path, so that the object can move through this path from 0s to 3s.

c) Twinkle effect

```
<animate attributeType="CSS" attributeName="opacity" from="1"
to="0" dur="1s" repeatCount="5" />
```

This sentence make the opacity change from 1 to 0 for 5 times, which can create twinkle effect.

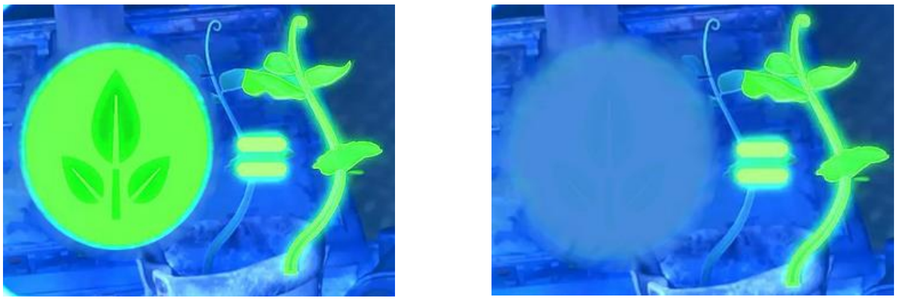


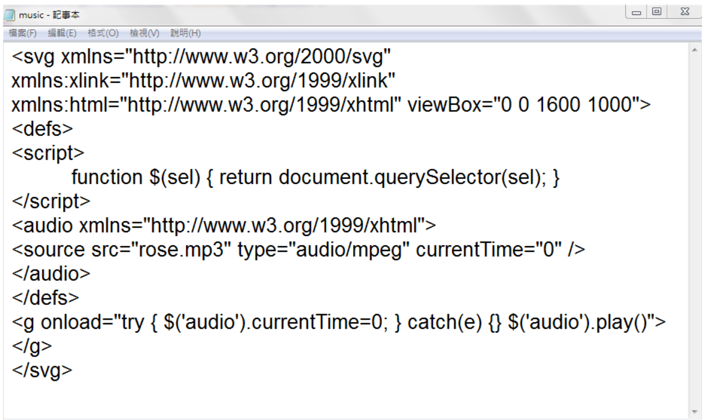
Figure 8 Twinkle effect

d) Add music:

In the main html file, I add the following sentence to embed music with the scenes.

```
<embed id="music" src="music.svg" type="image/svg+xml" />
```

Within the music.svg file, I written the following defines.



```
<svg xmlns="http://www.w3.org/2000/svg"
xmlns:xlink="http://www.w3.org/1999/xlink"
xmlns:html="http://www.w3.org/1999/xhtml" viewBox="0 0 1600 1000">
<defs>
<script>
        function $(sel) { return document.querySelector(sel); }
</script>
<audio xmlns="http://www.w3.org/1999/xhtml">
<source src="rose.mp3" type="audio/mpeg" currentTime="0" />
</audio>
</defs>
<g onload="try { $('audio').currentTime=0; } catch(e) {} $('audio').play()">
</g>
</svg>
```

Figure 9.The code in music.svg

When running the SVG comic in Chrome, every scene will be played with the background music.

e) Run all scene in a series

In the main HTML file, I write a function called myTimer to count the SVG files. Every SVG file is named in the form number.svg. So when running the main HTML file, every scene will be run.

```
document.getElementById("container").innerHTML="<embed  id='mySVG'
src='"+ i +".svg' type='image/svg+xml' />";
clearInterval(myVar);
myVar = setInterval(function(){myTimer()},scene[i-1]);
```

These sentences are called in a cycle to run a SVG file, clear it and run the next one.

5. Demonstration

This project can be displayed on Chrome Browser for about 1 minute. Each scene lasts for about 3 to 5 seconds. I try to tell the story about Wall-E and show the environment protection theme.

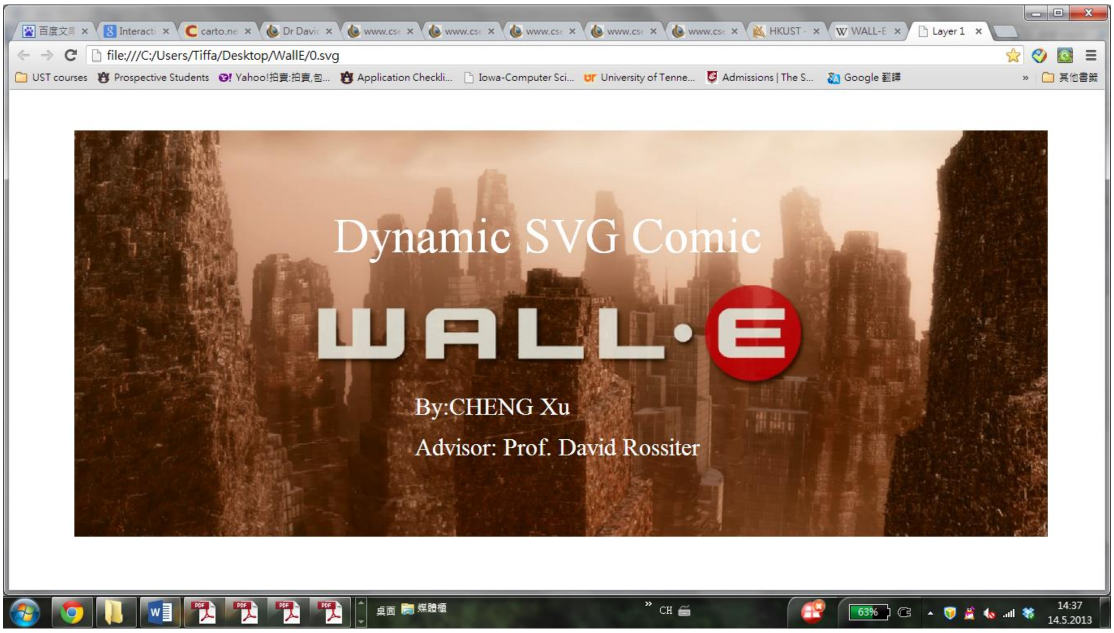

file:///C:/Users/Tiffa/Desktop/WallE/0.svg
Layer 1
Dynamic SVG Comic
WALL·E
By:CHENG Xu
Advisor: Prof. David Rossiter
14:37
14.5.2013

Figure 10. SVG scene with title in Chrome

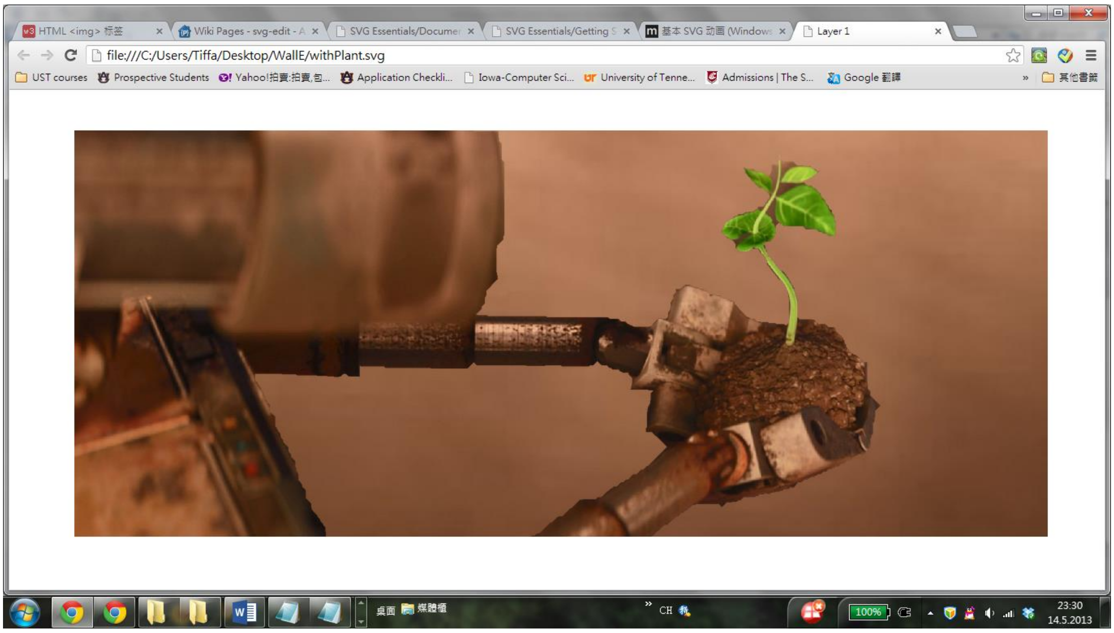


Figure 11. SVG scene

In the last scene, the robot can move together with the words.

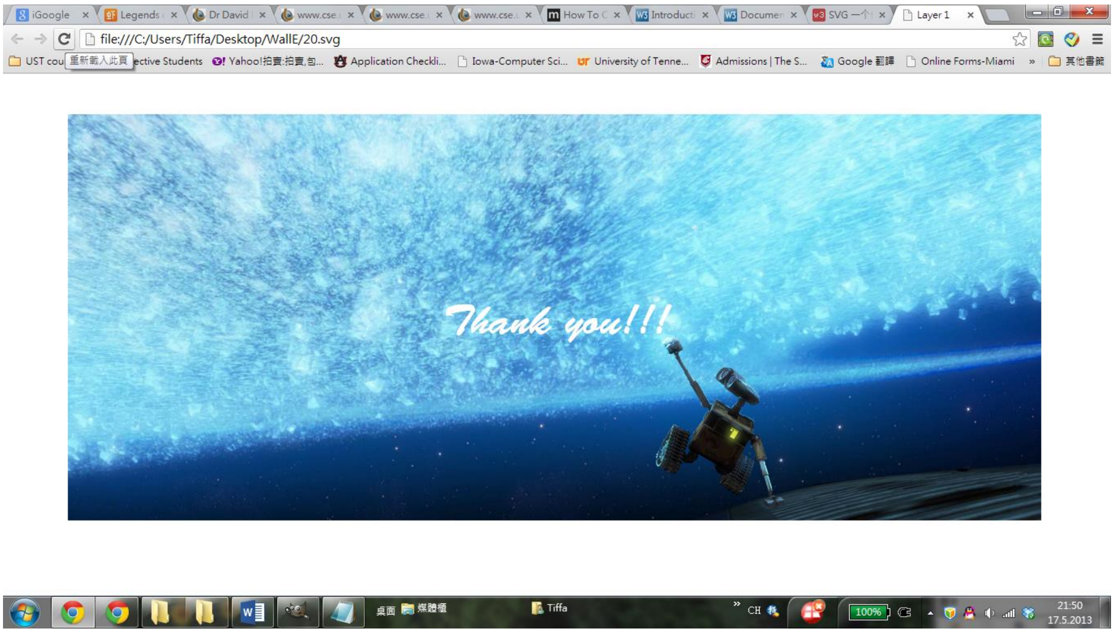

file:///C:/Users/Tiffa/Desktop/WallE/20.svg

Figure 10. Last scene

6. Further Development

This project can still be developed further. The most promising technology that I will apply is the scene switchover effects, which can make the scenes switch more naturally. What's more, if I have more time to learn html, JavaScript, I can add more animation effects. For example, 3D space effect, fluid effect. SVG has its advantages such as high definition, independence, and color control, however, I have not fully developed these advantages. This project can be developed in these aspects, like adding some high definition backgrounds creating by myself, adding more user interactive interfaces. I have watched a video on YouTube **[5]**. The lecturer used SVG to create some characters which can even have a conversation with the user. As far as I know, SVG provides 16 million colors palette which supports ICC color profile standard, RGB, gradient and mask. There still have much space to do further study in this area.

7. Conclusion

The XML based SVG realized a highly extendable two dimension image description and application in the form of plain text. The SVG itself can be used as animation form, it can also be combined with XML, JavaScript, SMIL, HTML, NWT and Java to create plenty of animations. This report introduced the application of SVG animation techniques in practical task. I explain the project structure and techniques in detail, and discussed several aspects that can be further developed.

This independent project provides me a good chance to apply what I learn from class into practice. I really appreciate the guidance of prof. Rossiter who not only teach me SVG programing in Computer Graphics course, but also direct and check my work frequently. Every time after we meet, I become very clear what is not good enough, how to make it better, and what I need to achieve. In this process, I learnt how to complete a project step by step, and really finished this interesting task. Although there are still many improvements I need to make, like enhancing my coding ability, adding more techniques in the projects, I'm very satisfied that I chose to do this independent project following my interest, and finished it successfully and willingly.

8. Reference

[1] SVG Introduction: http://www.w3.org/TR/SVG/intro.html

[2] DOM Introduction: http://www.w3.org/TR/WD-DOM/introduction.html

[3] SMIL3 Introduction: http://www.w3.org/TR/SMIL3/

[4] David Futato SVG Essentials February 2002: First Edition.

[5] YouTube Video: Journey through the Graphical Web

http://www.youtube.com/watch?v=P2Yj-8zGa3U

9. Appendix

Minutes of the 1st Project Meeting

Date: Thursday, 7 March 2013

Time: 10:30 AM

Place: Room 3512

Attending: Prof. Rossiter, CHENG Xu

Absent: None

Recorder: CHENG Xu

Approval of minutes

This is first formal group meeting, so there were no minutes to approve.

Report on Progress

CHENG Xu explained that she would make the SVG comic at the theme of Wall-E and displayed an example cartoon video which she will follow.

Discussion Items and Things to Do

- Learn how to animate with SVG
- Download the movie Wall-E and GIMP to edit the materials.

Meeting adjournment

The meeting was adjourned at 10:50 PM.

Minutes of the 2nd Project Meeting

Date: Wednesday, 5 April 2013

Time: 10:00 AM

Place: Room 3512

Attending: Prof. Rossiter, CHENG Xu

Absent: None

Recorder: CHENG Xu

Approval of minutes

The minutes of the last meeting were approved without amendment.

Report on Progress

CHENG Xu showed the first three scenes of the comic and discussed how to change scenes.

Discussion Items and Things to Do

- Speed up the progress
- Joint the scenes together

Meeting adjournment

The meeting was adjourned at 10:30 AM.

Minutes of the 3rd Project Meeting

Date: Monday, 15 April 2013

Time: 11:00 AM

Place: Room 3512

Attending: Prof. Rossiter, CHENG Xu

Absent: None

Recorder: CHENG Xu

Approval of minutes

The minutes of the last meeting were approved without amendment.

Report on Progress

CHENG Xu displayed the half-complete project.

Discussion Items and Things to Do

- Extend the comic duration to more than 1 minute.
- Apply more animation techniques to the project

Meeting adjournment

The meeting was adjourned at 11:20 AM.

Minutes of the 4th Project Meeting

Date: Thursday, 16 May 2013

Time: 11:00 AM

Place: Room 3512

Attending: Prof. Rossiter, CHENG Xu

Absent: None

Recorder: CHENG Xu

Approval of minutes

The minutes of the last meeting were approved without amendment.

Report on Progress

CHENG Xu displayed the SVG Comic. Prof. Rossiter gave some advice to improve the last scene and revise the report.

Discussion Items and Things to Do

- Revise the report
- Change the last scene of the SVG comic
- Hand in the report and final project materials in or before Saturday.

Meeting adjournment

The meeting was adjourned at 11:30 AM.