ფილტრების გამოყენება სცენარების ენაში JavaScript და პროგრამირების ენაში Java

ცაცა ნამჩევაძე

ქუთაისის წერეთლის სახელმწიფო უნივერსიტეტი

რ ე ზ ი უ მ ე

განხილულია ფილტრების გამოყენება სცენარების ენაში JavaScript და პროგრამირების ენაში Java გამოსახულების სხვადასხვა ვიზუალური ეფექტის მისაღებად. აღწერილია კონკრეტული ფილტრების დანიშნულება. ზოგიერთი ფილტრის შემთხვევაში მოყვანილია მაგალითები, რომელებზეც შედგენილია პროგრამული კოდები. ნაშრომის მიზანს წარმოადგენს მკითხველისათვის ფილტრებზე არსებული ინფორმაციის მიწოდება, ცოდნის შეძენა ფილტრების სწორად, მიზანმიმართულად გამოყენებისათვის. გაკეთებულია ანალიზი იმასთან დაკავშირებით, თუ რა შემთხვევაში არის მიზანშეწონილი გამოსახულების დამუშავებისათვის სცენარების ენის, პროგრამირების ენის, გრაფიკული რედაქტორების (მაგალითად: Adobe Photoshop, Macromedia Flash და სხვა) გამოყენება.

საკვანძო სიტყვები: სცენარების ენა JavaScript, პროგრამირების ენა Java, ფილტრი, alpha, revealtrans, basicimage, GropImageFilter, RGBImageFilter, Grayscale, Invert, Contrast, Sharpen.

1. შესავალი

ინტერნეტი თანდათან ჩვენი ცხოვრების განუყოფელი ნაწილი ხდება. მან უკვე მნიშვნელოვნად შეცვალა პლანეტის მოსახლეობის ყოველდღიური საქმიანობის სტილი, იგი არის უნიკალური და უსაზღვრო სამყარო, რომელსაც ადამიანი მყისიერად გადაყავს დედამიწის ნებისმიერ წერტილში. ინტერნეტ სამყაროში მომხმარებელი უამრავ საოცრებას ნახავს და ბევრ პრაქტიკულ საქმეს გააკეთებს. ინტერნეტის სხვსდასხვა მომსახურეობის ერთ-ერთ მნიშვნელოვან შემადგენელ კომპონენტს წარმოადგენს გამოსახულება, ამიტომ გამოსახულების დამუშავებასა და ვიზუალური ეფექტების მიღებას დიდი მნიშვნელობა აქვს ინტერნეტისათვის. ინტერნეტი განიცდის მუდმივ განვითარებას. მისი განვითარების კვალდაკვალ იქმნება ახალი საშუალებები გამოსახულებებთან სამუშაოდ.

2. ძირითადი ნაწილი

ნაშრომში განხილულია ფილტრების გამოყენება სცენარების ენაში JavaScript და პროგრამირების ენაში Java. ფილტრი წარმოადგენს ინსტრუმენტულ საშუალებას გამოსახულების გარდაქმნისათვის. სცენარების ენაში JavaScript ფილტრების გამოყენებით შეიძლება მივიღოთ საინტერესო ვიზუალური ეფექტები. მაგალითად: გამჭვირვალობის ხარისხის რეგულირება, სურათის თანდათანობით გამოჩენა (გაქრობა), ერთი გამოსახულების მეორე გამოსახულებით ნელ-ნელა შეცვლა და სხვა. ფილტრი შეიძლება გამოვიყენოთ არა მარტო გრაფიკული გამოსახულების მიმართ, ასევე სხვა ელემენტების მაგალითად: ტექსტების, ღილაკების მიმართაც.

სცენარების ენაში JavaScript შეიძლება გამოვიყენოთ შემდეგი ფილტრები: **alpha, reveal-trans, basicimage.**

გამჭვირვალობა. გრაფიკული ობიექტის გამჭვირვალობა შეიძლება ვცვალოთ **ფილტრის alpha** გამოყენებით 0-დან 100-მდე დიაპაზონში. მნიშვნელობა 0 შეესაბამება სრულ გამჭვირვალობას ე.ი. გამოსახულება იქნება უხილავი. მნიშვნელობა 100 შეესაბამება გამოსახულების სრულ გაუმჭვირვალობას. გარდა ამისა, გამჭვირვალობას აქვს გრადიენტული ფორმის რამოდენიმე ვარიანტი. მაგალითად, შეიძლება გამჭვირვალობა თანდათან გავზარდოთ ცენტრიდან გამოსახულების კიდემდე.

ფილტრის alpha, ისევე როგორც სხვა ფილტრების, გამოყენება ხდება სტილთა კასკადური ცხრილების საშუალებით შემდეგნაირად:

```
<HTML>
<STYLE>
  #myimage {position:absolute; top:20;left:40;
            filter:progid:DXImageTransform.Microsoft.alpha(opacity=8,  style=4)}
</STYLE>
 < IMG  ID="myimage"   SRC="car.jpg" >
</HTML>
```

თუ არ გამოვიყენებთ სიმბოლოს - #, მაშინ ფილტრი არ იმუშავებს.

პარამეტრი opacity განსაზღვრავს გამჭვირვალობის ხარისხს 0-100 დიაპაზონში.

პარამეტრი style იძლევა გამჭვირვალობის განაწილების გრადიენტს 0-დან 4-მდე გამოსახულების მიხედვით. თუ პარამეტრი style=0 ან არ არის მითითებული, მაშინ გრადიენტი არ გამოიყენება.

ფილტრს alpha აქვს სხვა პარამეტრებიც, რომლებიც განსაზღვრავს მართკუთხედის იმ არეს, რომლის მიმართაც გამოიყენება ფილტრი. გაჩუმების პრინციპით ფილტრი გამოიყენება მთელი გამოსახულების მიმართ.

სცენარში ფილტრის თვისებებზე მიმართვა ხდება შემდეგნაირად:

document.all.id_გამოსახულება.filters

["DXImageTransform.Microsoft.ფილტრის სახელი"]. პარამეტრი=მნიშვნელობა

მაგალითად: document.all.myimage.filters

["DXImageTransform.Microsoft.alpha"].opacity=50

ტრანსფორმაცია. გამოსახულების გარდაქმნის ფილტრების გამოყენებით შესაძლებელია გამოსახულების თანდათანობითი გამოჩენა (გაქრობა), აგრეთვე ერთი გრაფიკული ობიექტის ტრანსფორმაცია სხვა ობიექტად. მათ ეწოდებათ დინამიური ფილტრები. სტატიკური ფილტრებისაგან (მაგ. alpha) განსხვავებით დინამიური ფილტრების გამოყენება აუცილებლადაა დაკავშირებული სცენარებთან. გრაფიკული ობიექტის გარდაქმნის არსი მდგომარეობს იმაში, რომ თავდაპირველად

უნდა მოხდეს პირველი გამოსახულების დაფიქსირება, შემდეგ უნდა შეიცვალოს ეს გამოსახულება სხვა გამოსახულებით ან შეიცვალოს საწყისი გამოსახულების პარამეტრები. ყოველივე ამის შემდეგ უნდა შესრულდეს ტრანსფორმაცია. ზემოთ აღნიშნული ყველა მოქმედება სრულდება სცენარში. გამოსახულების ფიქსაცია და ტრანსფორმაცია წარმოებს სპეციალური მეთოდების გამოყენებით - შესაბამისად apply( ) და play( ). გარდაქმნის პროცესის შეწყვეტისათვის გამოიყენება მეთოდი stop( ).

გამოსახულების გარდაქმნა ხდება **ფილტრის revealtrans** საშუალებით. მას აქვს შემდეგი პარამეტრები:

duration - გარდაქმნის ხანგრძლივობა წამებში (მცურავმძიმიანი რიცხვი).

transition - გარდაქმნის ტიპი (მთელი რიცხვი 0-დან 23-მდე). ტიპს 23 შეესაბამება 0-დან 22-მდე ერთ-ერთი ტიპი, რომელიც არჩეულია შემთხვევითი გზით.

ქვემოთ მოცემულია პროგრამული კოდი გამოსახულების გარდაქმნაზე, რომელსაც ადგილი აქვს გრაფიკულ ობიექტზე დატკაცუნებისას. გამოსახულებაზე - ფაილი car.jpg დატკაცუნებისას მოხდება გარდაქმნა სურათად - ფაილი tulips.jpg. შემდგომი დატკაცუნებისას, პირიქით, მეორე გამოსახულება გარდაიქმნება პირველ გამოსახულებად.

```
<HTML>
<STYLE>
  #myimage {position:absolute; top:20;left:40;
        filter:progid:DXImageTransform.Microsoft.revealtrans(duration=5, transition=23)" >
</STYLE>
 < IMG ID="myimage" onclick="transform()"  SRC="car.jpg" >
<SCRIPT>
  function transform() {                              .
      document.all.myimage.filters("DXImageTransform.Microsoft.revealtrans").apply()
       if (document.all.myimage.src.indexOf("car.jpg")!=-1)
                                                    document.all.myimage.src= "tulips.jpg"
      else  document.all.myimage.src= "car.jpg"
  document.all.myimage.filters("DXImageTransform.Microsoft.revealtrans").play()
}
</SCRIPT>
</HTML>
```

<u>მობრუნება</u>. სკრიპტში შეიძლება გამოვიყენოთ **ფილტრი basicimage**. მისი საშუალებით შეიძლება გამოსახულება შემობრუნდეს 90^0-ის ჯერადი კუთხით. ფილტრი basicimage იღებს მთელ

მნიშვნელობებს: 0 (არ არის მობრუნება), 1 (90^0), 2 (180^0), 3 (270^0). ქვემოთ მოყვანილ პროგრამულ კოდში გამოსახულებაზე დატკაცუნებისას გამოსახულება შემობრუნდება 90^0-ით.

```
< HTML >
< STYLE >
        #myimage {filter:progid:DXImageTransform.Microsoft.basicimage}
</ STYLE >
< IMG ID ="myimage"  SRC="car.jpg"  onclick="rotor()">
< SCRIPT >
  function rotor() {
    var r=document.all.myimage.filters["DXImageTransform.Microsoft.basicimage"].rotation
        if (r==3)   r=0    else        r++
    document.all.myimage.filters["DXImageTransform.Microsoft.basicimage"] .rotation=r
 }
</ SCRIPT >
</ HTML >
```

ფილტრების გამოყენებისათვის პროგრამირების ენაში Java შემოტანილია პაკეტის java.awt.image კლასი ImageFilter. მას აქვს რამოდენიმე ქვეკლასი AreaAveragingScaleFilter, GropImageFilter, ReplicateScaleFilter და RGBImageFilter. გარდა ამისა, პაკეტში არის კლასი FilteredImageSource, რომლის საშუალებითაც ხდება ინტერფეისის ImageProducer რეალიზაცია. აღნიშნული კლასის ობიექტები ახდენენ გამოსახულების ფიქსელების ფილტრაციას, რომლებიც გამომუშავებულია კლასის ImageProducer მიერ.

ქვემოთ განხილულია ფილტრები: **CropImageFilter, RGBImageFilte, Grayscale, Invert**, **Contrast**, **Sharpen.**

ფილტრი CropImageFilter. იგი საწყისი გამოსახულებიდან ამოჭრის პატარა ზომის მართკუთხედის ფორმის არეებს. მისი გამოყენება მოსახერხებელია, როცა არ არის საჭირო დიდ მთლიან გამოსახულებებთან მუშაობა. თუ ქვეგამოსახულებებს აქვს ერთი და იგივე ზომა, მაშინ ფილტრის CropImageFilter გამოყენებით შესაძლებელია მათი გადაადგილება. ქვემოთ მოცემულია პროგრამული კოდი, რომლის შესრულების შედეგადაც გამოსახულება (პიკასოს მიერ შექმნილი სურათი) იყოფა 16 ერთნაირი ზომის მართკუთხედებად. შემდეგ ისინი გადაადგილდებიან შემთხვევითი გზით 32-ის ჯერადი გაცვლით.

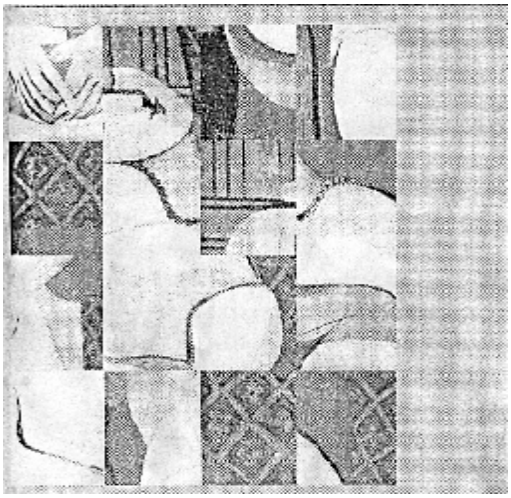
```
/* <applet code = TileImage.class width=288 height=399>
   <param name =img value=picasso.jpg>
   </applet>    */
import java.applet.*;
import java.awt.*;
```

```
import java.awt.image.*;

public class TileImage extends Applet {
   Image img;
   Image cell [ ] = new Image [4*4];
   Int   iw, ih,  tw, th;
public void init ( ) {
   try {
            img = getImage (getDocumentBase ( ), getParameter ("img"));
            MediaTracker t = new MediaTracker (this);
            t.addImage (img, 0);    t.waitForID(0);
            iw = img.getWidth(null);   ih = img.getHeight(null);
            tw = iw / 4;     th = ih /4;
            CropImageFilter f;
            FilteredImageSource fis;
            t = new MediaTracker (this);
            for (int y=0; y<4; y++)  {
               for (int x=0; x<4; x++)  {
                     f = new CropImageFilter (tw*x, th*y, tw, th);
                     fis = new FilteredImageSource (ing.getSource ( ), f);
                     int i = y*4+x;
                     cell[i] = createImage(fis);
                     t.addImage(cell[i], i);
            } }
   t.waitForAll ( );
   for (int i=0; i<32; i++)   {
            int si = (int) (Math. random ( ) * 16);
            int di = (int) (Math. random ( ) * 16);
            Image tmp = cell [si];   cell[si] = cell[di];   cell[di] = tmp;
            }
   } catch (InterruptedException e) { };
}
public void update (Graphics g)  {
   paint (g);
}
public void paint (Graphics g) {
```

```
for (int y=0; y<4; y++)  {
        for (int x=0; x<4; x++)  {
                g.drawImage(cell[y*4+x], x * tw, y * th, null);
} } } }
```

პროგრამის შესრულების შედეგი ნაჩვენებია 1-ელ ნახაზზე.



ნახ.1. აპლეტის TileImage გამოტანის მაგალითი

ქვემოთ მოკლედ არის აღწერილი Java-ში გამოყენებული ზოგიერთი ფილტრის დანიშნულება სანიმუშო პროგრამული კოდების გარეშე.

ფილტრი RGBImageFilter. იგი გამოიყენება გამოსახულების ფიქსელების ფერების ტრანსფორმირებისათვის. ამ ფილტრის საშუალებით შესაძლებელია გამოსახულების გაღიავება, კონტრასტის გაზრდა, აგრეთვე ფერადი გამოსახულების ნახევრადტონალურში გარდაქმნა.

ფილტრი Grayscale. ეს არის ფილტრის RGBImageFilter ქვეკლასი. ის იღებს წითელ, მწვანე და ლურჯი ფერების მნიშვნელობებს და გამოთვლის ფიქსელების სიკაშკაშეს. შემდეგ აბრუნებს ნაცრისფერ ფიქსელს, რომელსაც აქვს იგივე სიკავშკაშე, რაც ფერად ფერს.

ფილტრი Invert. ის იღებს ფიქსელებს წითელ, მწვანე და ლურჯი ფერებიდან. შემდეგ მოახდენს მათ ინვერსიას.

ფილტრი Contrast. იგი ძალიან გავს ფილტრს Grayscale. მის მიერ გამოყენებული ალგორითმის არსი მდგომარეობს იმაში, რომ კონტრასტის გაუმჯობესებისათვის იგი იღებს წითელ, მწვანე და ლურჯი ფერების მნიშვნელობებს ცალ-ცალკე. შემდეგ ამრავლებს 1,2-ზე, თუ მათი სიკავშკაშე მეტია 128-ზე. ხოლო თუ მათი სიკავშკაშე ნაკლებია 128-ზე, მაშინ მოხდება 1,2-ზე გაყოფა.

ფილტრი Sharpen. იგი არის აბსტრაქტული კლასის Convolver ქვეკლასი. ის გაირბენს ყოველ ფიქსელს გამოსახულების ფიქსელების საწყის მასივში imgpixels და გამოთვლის ფიქსე-

ლის ირგვლივ მაგ. 3x3 არეში სიკაშკაშის საშუალო მნიშვნელობას (მხედველობაში არ მიიღება ცენტრალური ფიქსელი). მიღებული მნიშვნელობებს მოათავსებს მასივში newimagepixels.

მაგალითად, თუ ფიქსელი არის 30-ჯერ უფრო ნათელი, ვიდრე მის ირგვლივ არე, მაშინ ასეთი ფილტრაცია დაახლოებით 30 მეზობელ ფიქსელს გახდის უფრო ნათელს. თუ ფიქსელი 10-ჯერ უფრო მუქია, ვიდრე მის ირგვლივ არე, მაშინ დაახლოებით 10 მეზობელი ფიქსელი გახდება უფრო მუქი.

ფილტრის Sharpen გამოყენების შედეგად გამოიკვეთება გამოსახულების საზღვრები, გლუვი არეები კი დარჩება უცვლელი.

Web - დიზაინერები გამოსახულებების დამუშავებისათვის აგრეთვე იყენებენ გრაფიკულ რედაქტორებს: Adobe Photoshop, Macromedia Flash და სხვა.

3. დასკვნა

სტატიაში დახასიათებულია სხვადასხვა სახის ფილტრები, რომლებიც გამოიყენება გამოსახულების ვიზუალური ეფექტების მისაღებად.

სცენარების ენაში JavaScript გამოყენებული ფილტრების პროგრამული კოდები უფრო მარტივია, ვიდრე პროგრამირების ენაში Java გამოყენებული ფილტრების კოდები. Java-ში გამოყენებულ ფილტრებს აქვს უფრო მეტი შესაძლებლობები გამოსახულების ვიზუალური ეფექტებისა და დამუშავების მაღალი ხარისხის მისაღებად, ვიდრე JavaScript-ში გამოყენებულ ფილტრებს. გამოსახულების დამუშავებისათვის გრაფიკული რედაქტორების გამოყენება მოითხოვს გარკვეული ჩვევების ცოდნას და ამასთან არის შრომატევადი.

გამოსახულების დამუშავების ზემოთ აღნიშნული საშუალებების ფონზე შეიძლება დავასკვნათ, რომ თუ გამოსახულებას არ ჭირდება რთული დამუშავება სასურველია გრაფიკული რედაქტორებისა და JavaScript-ის ფილტრების გამოყენება. იმ შემთხვევაში, თუ მოითხოვება გამოსახულების რთული დამუშავება და განსაკუთრებული ვიზუალური ეფექტების მიღება, მაშინ გამოვიყენებთ Java-ს ფილტრებს.

ლიტერატურა:

1. Ноутон П., Шилдт Г. Java, Санкт-Петербург. БХВ-Петербург. 2012
2. Бишоп Д. Java, Санкт-Петербург. Питер. 2012
3. Дунаев В. JavaScript. Санкт-Петербург. Питер. 2011
4. Холмогоров В. Основы Web-мастерства. Питер. 2011
5. Сырих Ю. Современный Web-дизайн. Москва- Санкт-Петербург-Киев. 2013,
exciton.es.rice.edu/Cpp/Resources.
6. codebeach.com
7. programmersheaven.com

USING FILTERS IN SCRIPTING LANGUAGE JAVASCRIPT AND PROGRAMMING LANGUAGE JAVA

Tsatsa Namchevadze
Kutaisi Tsereteli State University

Summary

In the article there is discussed using of filters in scripting language JavaScript and programming language Java for receiving different visual effects of image. There is described function of specific filters and for some filters there are given an examples upon which the program codes are made. The purpose of the article is to provide reader with information and acquire knowledge on filters and their correct using. In the article there is analyzed when it's advisable to use scripting language, programming language, graphic processors (for example: Adobe Photoshop, Macromedia Flash and other) for processing of image.

ПРИМЕНЕНИЕ ФИЛЬТРОВ В ЯЗЫКЕ СЦЕНАРИЕВ JAVASCRIPT И В ЯЗЫКЕ ПРОГРАММИРОВАНИЯ JAVA

Намчевадзе Ц.
Кутаиский государственный университет имени Церетели

Резюме

Рассмотрено применение фильтров в языке сценариев JavaScript и в языке программирования Java для получения различных визуальных эффектов изображения. Описано конкретное назначение фильтров. В случае некоторых фильтров приведены примеры, для которых составлены программные коды. Целью работы является донесение информации о фильтрах для читателей, приобретение знаний для правильного, целенаправленного применения фильтров. В связи с этим, в статье проведён анализ, в каких случаях целесообразно применение языка сценариев, языка программирования, графических редакторов (напр. Adope Photoshop, Macromedia Flash и др.) для разработки изображения.