SVG animacije grafičkih elemenata stvorenih u Adobe Illustratoru

Mandić, Anita

SVEUČILIŠTE U ZAGREBU
GRAFIČKI FAKULTET

ZAVRŠNI RAD

Anita Mandić

Sveučilište u Zagrebu
Grafički fakultet

Smjer: tehničko-tehnološki

ZAVRŠNI RAD

SVG ANIMACIJE GRAFIČKIH ELEMENATA STVORENIH U ADOBE ILLUSTRATORU

Mentor:
Doc. dr. sc. Tajana Koren Ivančević

Student:
Anita Mandić

Zagreb, 2016.

SAŽETAK

Tema podrazumijeva istraživanje mogućnosti primjene SVG animacija na grafičke elemente stvorene u vektorski baziranom računalnom program za crtanje, Adobe Illustratoru. SVG, ili punim nazivom *Scalable Vector Graphics*, je vektorski slikovni format baziran na XML-u. Koristi se za dvodimenzionalne slikovne prikaze podataka te podražava interaktivnost i animacije. Svi veći moderni internetski preglednici, kao što su Mozilla Firefox, Internet Explorer, Google Chrome, Opera, Safari i Microsoft Edge, posjeduju mogućnost renderiranja SVG koda, barem do neke razine. S obzirom da se SVG koristi sve češće na internetskim stranicama, tako se i navedeni preglednici trude pratiti navedeni trend.

Osim već spomenutog programa tvrtke Adobe Systems Incorporated, još nekoliko vektorski baziranih programa (uključujući CorelDRAW, Inkscape i Jasc WebDraw) omogućilo je spremanje datoteka u .svg formatu, no do danas je Adobe Illustrator ostao najrasprostranjeniji računalni program takve vrste. Iz toga će se razloga ovaj rad bazirati na stvaranju i uređivanju vektorskih grafičkih elemenata u Adobe Illustratoru, koji će se uporabom SVG koda naknadno animirati. U radu će također biti razrađeni konkretni primjeri te opis postupaka potrebnih za dobivanje istih.

Rad objedinjuje više područja grafičke tehnologije: dizajn, programiranje animacija te govori o prednostima animiranja programiranjem.

Ključne riječi: SVG, animacija, Adobe Illustrator

ABSTRACT

The topic connotes the research of all the possibilities of the application of SVG animations to graphic elements created in vector-based drawing software, Adobe Illustrator. Scalable Vector Graphics (SVG) is an XML-based vector image format. It has been used for two-dimensional graphics with support for interactivity and animation. All major modern web browsers - including Mozilla Firefox, Internet Explorer, Google Chrome, Opera, Safari, and Microsoft Edge - have at least some degree of SVG rendering support. Since websites have been using SVG images increasingly, the above-mentioned browsers are doing their best to catch up with the latest trend.

Aside from the already mentioned program by company Adobe Systems Incorporated , there are a couple of other vector-based programs (i.e. CorelDRAW, Inkscape i Jasc WebDraw) which have esured the storage of computer files in .svg format. However, to this day Adobe Illustrator has remained the most prevalent software of that kind. Thus, this paper will have a focus on the design and editing of vector graphic elements in Adobe Illustrator, that will become animated retrospectively, after the use of SVG code. The paper will also describe practical examples and the description of procedures for getting the practical examples.

The study includes several areas of graphic technology: design, animation programming and tells about the advantages of animations created by a software.

Key words: SVG, animation, Adobe Illustrator

Sadržaj

1. UVOD

Rad uključuje kreiranje vektorskih grafika u Adobe Illustratoru te spremanje istih u .svg formatu, nakon čega se na željene elemente primjenjuju SVG animacije. Opisuje se put od Adobe Illustratora do gotove animirane SVG grafike na internetskoj stranici te se analiziraju prednosti i nedostaci ovakve tehnike prikaza animacija na internetu u usporedbi s drugim dostupnim tehnikama. Cilj ovog rada je pokazati kako je moguće izvesti vizualno atraktivne animacije na *webu* korištenjem isključivo HTML i SVG tehnologija.

Na elemente i grupe elemenata primijenjene su animacije u obliku translacije, rotacije, mijenjanja veličine, mijenjanja boje i slično. Prilikom kreiranja objekata u Adobe Illustratoru potrebno je pravilno grupirati objekte te ih imenovati. Ovakav način prikaza animacije zauzima vrlo malo memorijskog prostora, radi čega datoteka ove animacije zauzima samo 1 MB memorijskog prostora.

2. TEORIJSKI DIO

2.1. Sustavi grafika

Dva osnovna sustava za prikaz grafičkih informacija na računalima su rasterska i vektorska grafika. Rastersku sliku je najlakše opisati kao niz točaka pri čemu je svaka opisana svojom odgovarajućom bojom, dok se vektorska grafika sastoji od točaka, rastezljivih linija i oblika koji se mogu naknadno micati i prilagođavati neovisno o pozadini i drugim grafičkim elementima na vektorskoj slici.

2.2.1. Rasterska grafika

U rasterskog grafici slika se sastoji od poredanih kvadratnih elemenata koji se nazivaju pikseli. Svaki piksel je zasebno definiran svojom vrijednosti za boju pomoću nekog od sustava boja ili kao indeksom iz liste boja. Drugi naziv za niz piksela je bitmapa, koja se često pohranjuje u sažeti format kako bi se smanjila veličina datoteke. Prilikom pokretanja slike, točnije prikaza grafike, program za pregled će prilikom otvaranja dekompresirati bitmapu i na taj je način prenijeti na zaslon uređaja.

Rasterska grafika je najpogodnija za fotografiju, s obzirom da se rijetko kada sastoji od vidljivih linija, krivulja ili pravilnih geometrijskih oblika. Skenirane slike se također spremaju kao bitmape. Postoji mnogo načina za sažimanje i spremanje rasterskih slika te prikaz istih. Od najčešćih formata tu su JPEG, GIF te PNG. Osim toga, alati za stvaranje i uređivanje rasterskih formata su raširenij te jednostavniji za korištenje od alata za crtanje vektorske grafike, stoga internetski preglednici su prvotno podržavali isključivo rastersku grafiku.

Kada zumiramo rastersku grafiku, pogram mora pronaći način da proširi svaki piksel. Najjednostavniji pristup zumiranju je množenje piksela, to jest povećavanje svakog piksela onoliko puta koliko povećavamo sliku. Tako će recimo program prilikom zumiranja 4 puta iz jednog piksela napraviti 4 ista, što će rezultirati vrlo zrnastom i nekvalitetnom slikom.

Ukoliko se autor slike ili ilustracije želi riješiti toga problema, može koristiti neke tehnike kao što su *edge-detection* ili *anti-aliasing*, koji će povećati kvalitetu uvećane slike, ne postoji garancija da će algoritam uspješno detektirati rubove oblika. Nadalje, za te je postupke potrebno uložiti mnogo vremena kako bi krajnji rezultat bio zadovoljavajući. [1]

2.2.2. Vektorska grafika

U vektorskoj grafici slika je opisana kao niz geometrijskih oblika definiranih pomoću odgovarajućih koordinata koje su mu dodijeljene. Dostupni geometrijski objekti su točke, linije, razni geometrijski oblici te poligoni. Na taj se način vektorski elementima može vrlo jednostavno i brzo naknadno mijenjati oblik i boja, bez utjecaja na kvalitetu prikaza na zaslonu uređaja. Isto vrijedi i za tekst, koji se može uređivati bez obzira je li transformiran ili pak animiran.

Vektorska se grafika najčešće koristi u *Computer Assisted Drafting* (CAD) programima koji se koriste za povećanu produktivnost dizajnera i kreiranje baza podataka za industrijske dijelove, što znači da je precizno mjerenje i mogućnost zumiranja u najmanje detalje vrlo bitno. Osim CAD programa, koriste se i programi za dizajniranje grafika koje su predviđene za tisak na tiskarskim strojevima visoke rezolucije, poput Adobe Illustratora, CorelDRAW-a i slično. Tu je i Adobe PostScript, grafički jezik kod kojeg se svaki lik opisuje linijama i krivuljama. Također se vrlo često koriste i Macromedia Flash sustavi.

Vektorska grafika je kodirana u binarnom formatu, stoga je internetskom pregledniku ili serveru vrlo teško analizirati binarni format i iz njega dinamično stvoriti vektorski grafički prikaz iz podataka.

Iako je vektorska grafika manje popularna od rasterske, posjeduje vrlo važno svojstvo koje nam rasterska grafika ne omogućuje, a to je konstantna kvaliteta prikaza slike bez obzira na zumiranje.

Dok se kod rasterske slike pikseli množe onoliko puta koliko se slika povećava, kod vektorske slike svaki objekt opisan je pomoću matematičkih izraza. Prilikom mijenjanja veličine objekata vektorske grafike, mijenjaju se matematičke značajke objekata, a ne sami objekti. To znači da program množi koordinate grafičkih elemenata onoliko puta koliko se slika povećava te uz pomoću novih dobivenih koordinata crta oblike u punoj rezoluciji. To omogućava zadržavanje jednake kvalitete kao i prije promjene, što daje glatke i prepoznatljive rubove linija i krivulja.

Pošto se zaslon ekrana sastoji od piksela, oba sustava grafika se naposlijetku pretvaraju u niz piksela kako bismo na zaslonu dobili odgovarajući prikaz. Kod

rasterske grafike slika se renderira pomoću stvaranja novih piksela, a kod vektorske grafike slika se renderira pomoću matematičke definicije objekata.

2.2. Vektorska grafika na *webu*

Postoje mnoge tehnologije koje su omogućile programerima i *web* dizajnerima razvijanje postupka prikazivanja crteža i ilustracija. Neke od njih su VML i SVG za 2D vektorsko crtanje, HTML5 Canvas za rastersko crtanje te WebGL za 3D grafičko renderiranje. [2]

SVG, punim nazivom *Scalable Vector Graphics*, je program baziran na XML tehnologiji, koji se pretežno koristi za vektorski slikovni prikaz podataka na internetskim stranicama. Omogućuje prezentaciju grafičkih informacija u sažetu i pokretnu formu. Koristiti za stvaranje potrebnih linija i oblika određivanjem koordinata točaka ili pak modifikacijom već postojećih rasterskih slika u vektorske crteže. Također je moguće kombinirati obje tehnike. Grafički elementi i atributi opisani u SVG-u također se mogu grupirati, stilizirati, transformirati, animirati te složiti unutar prethodno prikazanih elemenata. Osim toga na elementima se mogu aplicirati alfa maske, predlošci te mnoge druge opcije.

Zanimanje za SVG drastično raste te alati za stvaranje i pregled SVG datoteka su široko dostupne. Stoga se pojavila potreba za unaprijeđenjem internetskih preglednika kako bi isti mogli prikazati grafičke elemente stvorene u SVG formatu. Razlog tomu je sve veća količina podataka te uređaji s velikim mogućnostima prikaza zahtjevnih grafičkih elemenata, koji su u stanju vizualizirati potrebne umrežene podatke. Iz tih su razloga internetski preglednici od jednostavne početne točke u svijet informacija prerasle u vrlo interaktivan medij.

Kako bi se pospješilo i olakšalo korištenje SVG-a, neki vektorski grafički programi dodali su mogućnost uvoza i izvoza crteža u SVG format, te je time SVG postao standardni format za razmjenu informacija. Među njima su Adobe Illustrator, Jasc WebDraw, CorelDRAW i Inkscape. Pošto je SVG dio XML tehnologije, također radi zajedno s drugim jezicima iste tehnologije.

Za razliku od VML-a, konkurentnog vektorskog programskog jezika, SVG ne podržava odvajanje poretka crteža od poretka dokumenta za preklapajuće elemente. Također, tekst kao element unutar SVG-a možemo koristiti u svim

XML varijablama unutar SVG koda, a to nam pospješuje pristupačnost pojedinih SVG grafičkih elemenata.

Primjena i provedba SVG-a koja je trenutno dostupna korisnicima na internetskim preglednicima nije toliko napredna kao što je to slučaj s nekim konkurentskim tehnologijama.

SVG osigurava grafičke elemente poput krugova, pravokutnika, krivulja i poligona. Korijen svake SVG datoteke je element <*svg*>. Osim gore navedenih oblika, moguće je definirati i stil svakog objekta, grupirati oblike s elementom <*g*> te ih je moguće animirati.

Iako se SVG oslanja na relativno malom skupu elemenata koji tvore grafički prikaz, on nudi mogućnosti za animaciju i primjenu različitih filtera. Unatoč tome, nešto je ograničeniji od JavaScript-a ili CSS-a, kod kojih je nešto lakše manipulirati objektima. No i sa SVG-om je moguće proizvoljno postići skoro pa jednako zahtjevne animacije.

SVG je podržan u svim modernim internetskim preglednicima. Svakom novom verzijom internetski preglednici dodaju nova podržana svojstva kako bi korisničko iskustvo bilo što bolje.

Za razliku od HTML-a, SVG elementi su osjetljivi na razliku kurenta i verzala, stoga je potrebno pri pisanju istih paziti na oblik teksta koji se koristi. Nadalje, atributske vrijednosti u SVG-u obavezno se pišu unutar navodnika, makar je riječ o brojkama.

Dodaci za internetske preglednike, koji služe za prikaz SVG-a na zaslonu uređaja, nude slične mogućnosti kao i Flash. Pošto su SVG datoteke zapravo XML, tekst SVG datoteka dostupan je svim preglednicima koji mogu analizirati XML.

S obzirom da SVG datoteka, čijim se kodom prikazuju vektorski oblici, sadrži mnoge dijelove teksta koji se ponavljaju veliki broj puta, istu je moguće i sažeti pomoću algoritama. Pritom se ne gube podatci. Ukoliko se SVG slika kompresira pomoću standardnog gzip algoritma, za nju se koristi naziv SVGZ slika te dobiva ekstenziju .svgz. Veličina SVGZ datoteka najčešće je 20 do 50 posto manja od veličine originalne SVG datoteke.

2.3. Povijest SVG-a

1998. godine *SVG Working Group*, sastavnica World *Wide Web Consortiuma*, glavne međunarodne organizacije za standarde za *World Wide Web*, počela je raditi na stvaranju SVG standarda. Predstavnik grupe bio je Chris Lilley, računalni znanstvenik koji je također bio član grupe zaslužne za razvoj HTML-a 2.0 te je jedan od autora PNG formata za rastersku grafiku.

Godinu ranije završeni su i konkurentni standardi Precision Graphics Markup Language, skraćeno PGML, koji je razvijen iz PostScripta, te Vector Markup Language, skraćeno VML, razvijen iz Microsoft-ovog RTF-a. Postojali su određeni problemi u oba konkurentska formata, stoga je bilo potrebno osmisliti novi format koji će pokriti i ključne nedostatke istih. Stoga su pri stvaranju SVG standarda oba prethodno navedena standarda znatno utjecala na proces te završni dizajn SVG-a. Iako je SVG relativno star, prihvaćanje i podrška SVG-a od strane internetskih preglednika još nije potpuna te teče vrlo sporo.

4. rujna 2001. godine prva verzija SVG-a, SVG 1.0, postao W3C prijedlog. Druga verzija, SVG 1.1 izlazi dvije godine kasnije. Nije se mnogo razlikovala od prvotne verzije. Jedna od ključnih razlika bila je modulacija podvrsta u definirane profile.

Tada se već razvila potreba i za SVG profilima za mobilne uređaje, stoga su razvijene verzije *SVG Tiny* (SVGT) i *SVG Basic* (SVGB), koji su također predloženi kao standard od strane W3C-a. Pošto su predviđeni za uređaje s ograničenim mogućnostima prikaza, tako SVGT ne podržava stilizaciju i ispisivanje naredbi, a koristi se isključivo za lošije mobilne uređaje, dok SVGB nudi više mogućnosti, a predviđen je za jače mobilne uređaje poput *smartphone* uređaja. Pojačana verzija *SVG Tinyja,* nazvana *SVG Tiny 1.2*, kasnije je postala autonomna verzija. [3]

2003. godine *The 3rd Generation Partnership Project* (*3GPP*), organizacija koja udružuje telekomunikacijska društva, prihvatila je *SVG Tiny* kao glavni vektorski grafički format za buduće mobilne uređaje. SVGB je tada proglašen neobavezan, no kasnije je dodan u traženi format za prikaz grafičkih podataka u 3GPP-u. Ni jedan mobilni profil ne uključuje potporu za potpuni DOM, dok samo *SVG Basic* ima mogućnost ispisa naredbi na mobilnim uređajima.

MPEG-4 Part 20 standard - Lightweight Application Scene Representation (LASeR) i *Simple Aggregation Format* (SAF) koriste *SVG Tiny* kao osnovu, no poboljšane su ključne mogućnosti koje prilagođavaju prikaz na mobilnim uređajima. Neki od tih svojstava su dinamična ažuriranja, binarno šifriranje te prikaz fontova.

2008. godine, pet godina nakon prethodne verzije SVG-a, počeli su raditi na *SVG Full 1.2* verziji. Razvijanje iste ubrzo je prekinuto radi boljeg i naprednijeg SVG-a 2, no ipak su se vratili dovršavanju istog kao samostalnom i odvojenom propisu. *SVG 1.1 Second Edition*, prošireno izdanje SVG-a 1.1, objavljen je 16. kolovoza 2011. godine. Uključuje isključivo dodatna razjašnjavanja, a ne nova svojstva.

SVG 2 izašao je krajem 2015. godine, te se znatno razlikuje od dosadašnjih verzija. Omogućuje bolju integraciju u druge standarde korištene za internetske stranice, kao što su HTML5, CSS i JavaScript.

2.4. Prednosti SVG-a

Glavna prednost SVG-a je u tome što se SVG može uređivati u bilo kojem uređivaču teksta (Notepad, Notepad++, Office Word, Adobe Dreamweaver i slično). SVG slike moguće je pretraživati, indeksirati, sažimati i ispisivati.

Tisak vektorskih grafika stvorenih pomoću SVG standarda je vrlo jednostavno te ne dolazi do problema poput ispisa rasterskih grafika radi gubitka kvalitete zbog rezolucije.

Osim SVG-a, glavni način stvaranja i prikaza vektorske grafike na *webu* je Flash, no SVG ima veliku prednost naspram Flasha, a to je što je sukladan s drugim standardima poput XSL-a i DOM-a. Također, Flash se ne oslanja na „open source“ tehnologiju. [4]

SVG se ponovno sve više upotrebljava na internetskim stranicama te dizajneri i razvojni programeri sve češće pribjegavaju navedenom formatu za prikaz manjih ikona, linijske grafike, logotipova te jednostavnijih animacija na istima, koji se učestalo ponavljaju na internetskoj stranici. Nadalje veličina prosječne SVG datoteke je znatno manja u usporedbi s drugim formatima poput PNG-a. To omogućuje brže učitavanje stranice i elemenata na njoj.

Osim navedenog, SVG također uključuje stranični opis poput PDF-a. Nudi bogatu grafiku te je sukladan sa CSS-om za lakše i bolje stiliziranje. SVG može postaviti svaki slovni znak i sliku na odabranu lokaciju na otisnutoj stranici. Specifična funkcionalnost koja je pomogla u realizaciji printanja pomoću SVG-a uvedena je u SVG 1.2.

SVG grafičke elemente moguće je dodati u *Extensible Stylesheet Language Formatting Objects* (XSL-FO) datoteku. Zatim se ona može konvertirati u Adobe PDF format. Na taj se način može dobiti visokokvalitetna priprema za tisak te na kraju i sam otisak grafike. SVG se često koristi u mnogim granama znanosti, ne isključivo samo za web. Neki od primjera su autori geografskih karata te meteorolozi. SVG im nudi prenosiv format visoke kvalitete, koji je sposoban prenijeti mnogo detalja.

2.5. Stvaranje grafičkih elemenata u SVG-u

2.5.1. Definiranje osnovne strukture SVG dokumenta

Osnovna struktura SVG dokumenta sastoji se od korijena <*svg*> u kojeg se pozicioniraju ostali dijelovi koda, poput osnovnih oblika te njihovih karakteristika. Korijen <svg> definira širinu i visinu završne grafike. Osim <*svg*> elementa, na početku SVG datoteke dodaju se još neki bitni elementi, poput <*title*> elementa, koji određuje naslov samog dokumenta, te <*desc*>, koji omogućuje potpuni opis grafike.

```
<svg width="800" height="500">
     <title>Naslov</title>
     <desc>Opis grafike</desc>
     <!—elementi koji tvore grafiku-—>
</svg>
```

Iz navedenog primjera, vidljivo je da je SVG dokument definiran širinom od 800 piksela i visinom od 500 piksela, nakon čega je uveden naslov i opis, poslije čega se ubacuju potrebni elementi koji definiraju crtež zajedno sa stilizacijom pojedinih elemenata. Također, tekst unutar <!— —> oznake je komentar koji se može dodati bilo gdje u SVG dokumentu.

Jednom kad se definira visina i širina unutar <*svg*> oznake, mogu se početi crtati osnovni oblici. Najčešći oblici su linije, pravokutnici, poligoni, krugovi i elipse. Za crtanje navedenih oblika potrebno je poznavanje sintakse kojima se svaki definira. Svaka SVG oznaka pridružena je odgovarajućem obliku. Neki oblici mogu se crtati pomoću drugih oblika, no potrebno je izabrati odgovarajuću oznaku kako bi kod SVG dokumenta bio što kraći i pregledniji.

2.5.2. Osnovni oblici

2.5.2.1 Linije

Ravne crte unutar SVG-a definiraju se pomoći <*line*> taga, pri čemu je potrebno odrediti x i y koordinate početne i krajnje točke linije. Koordinate mogu biti bez navedene mjerne jedinice ili s mjernim jedinicama kao što su četverac (em), inč (in) i mnogim drugima.

Osnovna struktura linije u SVG-u izgleda ovako:

```
<line x1="start-x" y1="start-y" x2="end-x" y2="end-y">,
```

pri čemu *x1* i *y2* definiraju koordinate prve točke, a *x2* i *y2* koordinate druge točke. Primjer jedne takve linije prikazan je na Slici 1.

Slika 1. Linija

Linija sa Slike 1 opisana je SVG elementom:

```
<line stroke="black" x1="100" y1="100" x2="500" y2="100"/>
```

2.5.2.2 Pravokutnici

Za kreiranje pravokutnika najčešće se koristi <*rect*> element. Pri tome koriste se šest osnovnih atributa za kontrolu nad oblikom, točnije pozicioniranje i definiranje istog, a to su *x*, *y*, *width*, *height*, *rx* i *ry*. Atributi *x* i *y* određuju koordinate gornjeg lijevog vrha pravokutnika, točnije koliko je ta točka udaljena od početne lijeve gornje točke SVG dokumenta. Atribut *width* odnosi se na širinu pravokutnika, a *height* na visinu pravokutnika. Atributi *rx* i *ry* se mogu, ali i ne moraju koristiti. Ukoliko željeni pravokutnik ima zaobljene rubove, tada se definiraju i navedeni atributi. Ukoliko nisu definirani, njihova vrijednost je 0, što znači da linije koje omeđuju vrhove tvore kut od 90°.

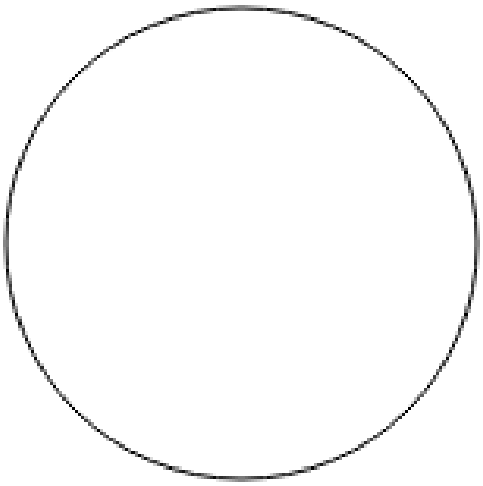
Slika 2. Pravokutnik

Slika 2 prikazuje pravokutnik, koji je definiran na sljedeći način:

```
<rect fill="none" stroke="black" x="100" y="100"
width="200" height="100" rx="10" ry="10"/>
```

2.5.2.3 Krugovi

Ukoliko je potrebno nacrtati krug unutar SVG dokumenta, definiramo ga pomoću <*circle*> taga, uz kojeg također koristimo neke atribute, a to su cx, cy i r. Cx određuje apscisu, a cy ordinatu središta kruga, dok se pomoću r atributa definira radijus kruga.



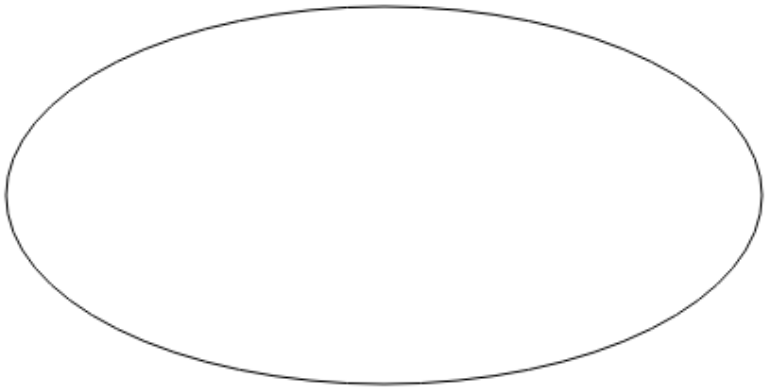
Slika 3. Krug

Primjer kruga nalazi se na Slici 3, a stvoren je pomoću sljedećeg elementa:

```
<circle fill="none" stroke="black" cx="100" cy="100"
r="75"/>
```

2.5.2.4 Elipse

Elipsama je potrebno nešto više atributa u usporedbi s krugom, kako bi se definirale dvije poluosi, velika i mala. Parametri elipse definiraju se pomoću cx i cy atributa, koji se odnose na pozicioniranje središta elipse, te rx i ry koji govore o dužini prethodno navedenih poluosi.



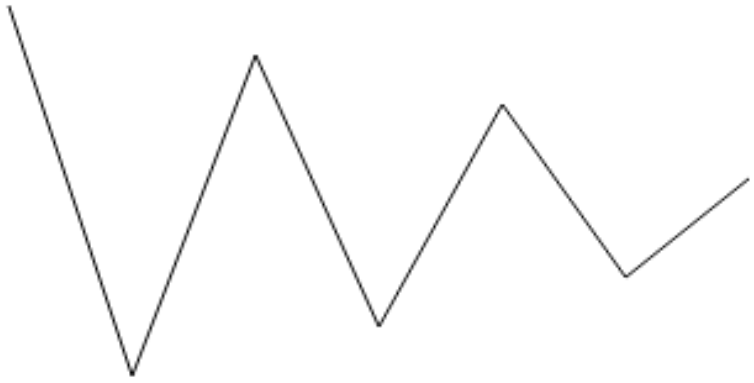
Slika 4. Elipsa

Elipsa sa Slike 4 opisana je pomoću navedenog koda:

```
<ellipse fill="none" stroke="black" cx="250" cy="250"
rx="200" ry="100"/>
```

2.5.2.5 Polilinije

Polilinija je zapravo skup povezanih ravnih linija. Ponekad broj tih linija je vrlo velik, stoga se x i y koordinate ne definiraju odvojeno svaka svojim atributom, već se koordinate točaka definiraju unutar points atributa. Svaka x i y točke koordinate i dalje se definira pomoću pripadajućih brojki.



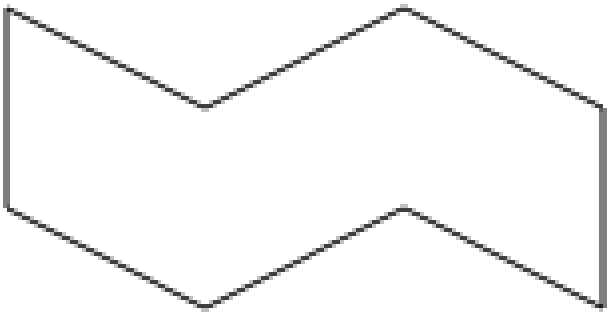
Slika 5. Polilinija

Polilinija prikazana na Slici 5 definirana je pomoću navedenog polyine taga i pripadajućih točaka:

```
<polyline points="50 50, 100 200, 150 70, 200 180, 250 90, 300 160, 350 120"/>
```

2.5.2.6 Poligoni

Poligon je definiran na isti način kao i polilinija, s obzirom da je također sastavljen od niza ravnih linija, no u ovom slučaju one tvore neki zatvoreni oblik. Poligon automatski spaja početnu i krajnju definiranu točku.



Slika 6. Poligon

Poligon na Slici 6 definiran je na sljedeći način:

```
<polygon stroke="black" fill="none" points="50 150, 100 175, 150 150, 200 175, 200 225, 150 200, 100 225, 50 200"/>
```

2.5.2.7 Putanje

Putanja je zapravo najčešći i najutjecajniji oblik koji se koristi, a definira se pomoću <*path*> taga. Razlog tomu je što nudi različite mogućnosti oblikovanja, što znači da pomoću putanje moguće je dobiti jednostavne, ali i vrlo kompleksne oblike. Tako se pomoću putanje mogu crtati i prethodno navedeni oblici, ali i razne krivulje, kružni isječci i drugo. Putanja sadrži samo atribut *d*. Koordinate unutar d atributa uvijek se pišu bez mjernih jedinica.

Putanja stvara kompleksne oblike udruživanjem brojnih ravnih i zakrivljenih linija. Ako oblik sadrži isključivo ravne linije, može se crtati pomoću putanje ili

poligona, no ukoliko poželjni oblik ima barem jedan zaobljeni dio, tada je svakako poželjno koristiti putanju za crtanje oblika.

Atribut d, koji definira putanju, sastoji se od mnoštva naredbi i parametara koje te naredbe koriste. Svakoj naredbi za putanju dodijeljeno je odgovarajuće slovo. Tako postoji *M* za move to naredbu, *L* za line to *H* za horizontalnu liniju, V za vertikalnu, *C* za Bezierovu krivulju, *Q* za kvadratičnu Bezierovu krivulju i tako dalje.

Također postoji razlika između kurenta i verzala. Svaka naredba dolazi u oba oblika, no verzali se odnose na apsolutno pozicioniranje, točnije u odnosu na središte koordinatnog sustava cijelog SVG dokumenta, dok se kurenti odnose na relativno pozicioniranje, odnosno pozicioniranje nove točke u odnosu na zadnju prethodno definiranu točku.

Tako naprimjer navedena naredba tvori Bezierovu krivulju iz Slike 7.

Slika 7. Putanja

```
<path d="M50 50 C 50 300, 300 -100, 500 100" />
```

2.5.3. Tekst

Ukoliko je u SVG dokument potrebno ubaciti tekst, tada se to postiže ubacivanjem <*text*> elementa unutar <*svg*> korijena. Tekst je potrebno definirati uz pomoć dva glavna atributa, *x* i *y*, koji određuju gdje će se tekst pozicionirati. Atribut *text-anchor* može imati vrijednosti *start*, *middle*, *end* i *inherit*, što omogućava manipulaciju smjera ispisa teksta u odnosu na točku definiranu x i y koordinatama.

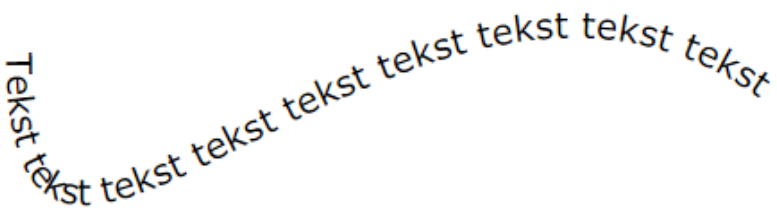
Slika 8. Tekst

```
<text font-family="Verdana" font-size="55" x="100" y="100">
     Ovdje  ide tekst!
</text>
```

Gore navedeni kod tvori tekst sa Slike 8. Bitno je napomenuti da je neophodna sastavnica opisivanja teksta sam font u kojem će se tekst prikazati. Font i druga bitna obilježja prikaza teksta definiraju se sljedećim elementima: : *font-family*, *font-style, font-weight, font-variant, font-stretch, font-size, font-size-adjust, kerning, letter-spacing, word-spacing* i *text-decoration*.

Kao i osnovni oblici, tekstu se može dodijeliti boja obruba i ispune, ali i gradijenti i teksture istih, što omogućava jednostavnije i snažnije stiliziranje teksta u odnosu na CSS 2.1.

Tekst se također može protezati duž neke putanje, što se postiže uz pomoć <*textPath*> taga. Prethodno je potrebno programirati putanju i dodijeliti joj identitet, u ovom slučaju pod nazivom „putanja“. Zatim xlink:href, atribut <*textPath*> elementa, omogućava prepoznavanje željene putanje po kojoj će se ispisati tekst.



Slika 9. Tekst na putanji

Prikazani tekst na putanji iz Slike 9 opisan je u SVG-u pomoću definirane putanje i teksta:

```
<path fill="none" id="putanja" d="M50 50 C 50 300, 300 -
100, 500 100"/>
<text font-family="Verdana" font-size="20">
     <textPath xlink:href="#putanja">
          Tekst tekst tekst tekst tekst
          tekst tekst tekst tekst
     </textPath>
</text>
```

2.6. Stiliziranje grafičkih elemenata

Bez primjene stilizacije, objekti su standardno crni. Stiliziranje omogućuje programerima i dizajnerima dizajniranje elemenata na zanimljiv i učinkovit način. Kada ne bi postojala raznolika svojstva za stilizaciju elemenata koje nudi SVG, programerima i dizajnerima bilo bi mnogo teže stvoriti toliko zanimljive vektorske grafike na internetskim stranicama.

2.6.1. Obrub i ispuna

Dva osnovna atributa pomoću kojeg se može obojati SVG element su *fill* za boju ispune i *stroke* za boju obruba. Boja se definira na iste načine kao i kod HTML-a, a to su pomoću imena boja, RGB vrijednosti, heksadekadskom vrijednosti i slično.

Nadalje moguće je specificirati i prozirnost pomoću atributa *fill-opacity* te *stroke-opacity*. Njihova vrijednost je unutar spektra od 0 do 1, pri čemu 1 označava potpunu neprozirnost, a 0 potpunu prozirnost.

Osim boje i prozirnosti, obrubu je moguće definirati način na koji će se iscrtavati rubovi linija. To se postiže atributom *stoke-linecap*, čija vrijednost može biti *butt*, *square*, *round* i *inherit*. Ukoliko ovaj atribut nije naznačen, primijenjuje se *butt* vrijednost.

Stroke-width svojstvo određuje kolika će biti širina obruba. Obrubi se crtaju oko putanje ravnomjerno s obje strane.

Prilikom stiliziranja polilinije ili poligona, upotrebom atributa *stroke-linejoin* određuje se kako će se spajati vrhovi dva linijska segmenta. Ukoliko se koristi vrijednost *miter*, tada je vrh vrlo oštar. Kod vrijednosti *round*, vrh je zaobljen, dok kod vrijednosti *bevel* vrh je odsiječen.

Također je moguće nacrtati isprekidane linije. Takav efekt postiže se atributom *stoke-dasharray*. Prvi broj određuje dužinu iscrtanog područja, a drugi razmaka između iscrtanih dijelova linije. Osim toga može se dodati još vrijednosti, pri čemu i dalje vrijedi da prva brojka označava iscrtani dio, druga prazni, treća iscrtani i tako dalje.

Bitno je još spomenuti i ostala svojstva. *Fill-rule* određuje kako će se ispuniti oblici bojom u slučaju ako se preklapaju. *Stroke-dashoffset* određuje gdje će počinjati isprekidanost linije.

2.6.2. Gradijenti

Osim vrlo jednostavnih atributa, postoje i neki kompleksniji načini na koji možemo ispuniti i iscrtati oblike unutar SVG grafike, a to su gradijenti i uzorci.

Postoje dva tipa gradijenata, a to su kružni i pravocrtni. Gradijenti se definiraju unutar <*defs*> taga kako bi se mogli koristiti više puta.

Za dodavanje pravocrtnog gradijenta, koristi se <*linearGradient*> element, koji se smješta unutar <*defs*> taga. Unutar samog <*linearGradient*> taga, koriste se <*stop*> tagovi koji određuju offset, boju i prozirnost ispune na tim postocima i slično. Orijentacija gradijenta kontrolira se preko dvije točke, koje označuju početak i kraj gradijenta. Bez navedenih koordinata tih točaka, koje se označavaju s *x1*, *y1*, *x2* i *y2*, gradijent se širi horizontalno. Za rotaciju potrebno je prilagoditi navedene koordinate.

Jedan takav primjer vidljiv je na Slici 10.

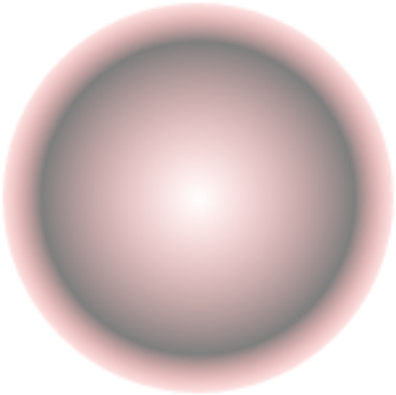
Slika 10. Pravocrtni gradijent kao ispuna kruga

```
<defs>
     <linearGradient id="G1" x1="0" x2="0" y1="0" y2="1">
          <stop offset="0% " stop-color="red"
          stop-opacity="0"/>
          <stop offset="50%" stop-color="black"
          stop-opacity="0.5"/>
          <stop offset="100%" stop-color="green"
```

```
            stop-opacity="0"/>
        </linearGradient>
</defs>
<circle fill="url(#G1)" cx="100" cy="100" r="75"/>
```

Kako bismo primijenili definirani gradijent, vrlo je bitno dati mu *id* atribut. U ovom primjeru, *id* gradijenta je „G1“, stoga se kasnije njega poziva preko *fill*-a ili *stroke*-a objekta kojem želimo dodati navedeni gradijent, u ovom slučaju *url(#G).*

Na sličan način funkcionira i kružni gradijent. Za njegovu primjenu koristi se tag <*radialGradient*>. Za razliku od pravocrtnog gradijenta, kružni definiramo pomoću *cx* i *cy*, te *fx* i *fy* koordinatne točke. Atributi c*x* i *cy* označavaju centralnu točku gradijenta, a *fx* i *fy* fokusnu točku. [6] Navedeni primjer koda tvori kružni radijent primijenjen na krug na Slici 11.



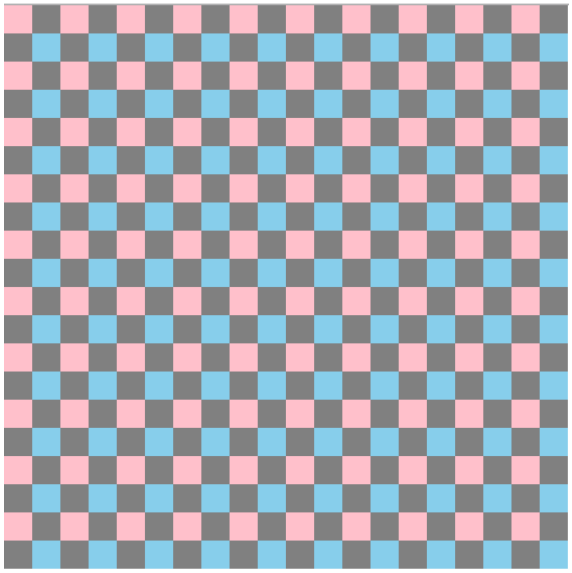
Slika 11. Kružni gradijent kao ispuna kruga

```
<defs>
     <radialGradient id="G2" cx="0.5" cy="0.5" r="0.5"
     fx="0.5" fy="0.5">
          <stop offset="0%" style="stop-color:red;
          stop-opacity:0" />
          <stop offset="80%" style="stop-color:black;
           stop-opacity:0.5" />
          <stop offset="100%" style="stop-color:red;
          stop-opacity:0.1" />
    </radialGradient>
</defs>

<circle fill="url(#G2)" cx="100" cy="100" r="75"/>
```

2.6.3. Uzorci

Kao i gradijenti, uzorci se nalaze u <*defs*> elementu SVG dokumenta. Uzorak može sadržavati bilo koje prethodno navedene i stilizirane oblike. Obavezno je definirati veličinu uzorka te osnovne elemente od kojih se sam uzorak sastoji.

Uzorci imaju atribut *patternContentUnits*, koji određuje u kojim jedinicama su opisani elementi unutar *pattern* elementa. Ukoliko se ne navede drugačija vrijednost za ovaj element, koordinatni sustav objekata unutar uzorka potpuno se razlikuje od koordinatnog sustava samog *pattern* elementa.Ukoliko su nacrtani objekti izvan definiranog područja veličine uzorka, tada se oni neće prikazivati. [6]



Slika 12. Uzorak kao ispuna pravokutnika

Uzorak sa Slike 12 sastoji se od 3 elementa.

```
<defs>
     <pattern id="Uzorak" x="0" y="0" width=".1"
     height=".1">
          <rect x="0" y="0" width="50" height="50"
          fill="grey"/>
          <rect x="0" y="0" width="25" height="25"
          fill="pink"/>
          <rect x="25" y="25" width="25" height="25"
          fill="skyblue"/>
    </pattern>
</defs>
<rect fill="url(#Uzorak)" width="500" height="500"/>
```

2.7. Animacije grafičkih elemenata

Prilikom stvaranja SVG-a, W3C nije htio ograničiti navedeni standard, koji bi stvarao isključivo nepokretne vektorske grafike. Stoga je osmišljen SMIL, dio SVG-a koji statičnim slikama daje „život“ dodijeljivanjem raznolikih animacija na pojedine elemente istih. Zbog tog su razloga animacije vrlo značajne i često primijenjene na *web* stranicama.

SVG animacije omogućuju pomicanje elementa duž putanje, animiranje transformacija, boje i drugih osnovnih atributivnih vrijednosti tranzicija.

SVG nudi pet osnovnih elemenata za animacije, a to su:

1. <*animate*>, koji omogućuje prilagodbu vrijednosti skalarnih atributa u jediniciama vremena
2. <*animateMotion*>, koji omogućuje pomicanje elementa duž putanje
3. <*animateTransform*>, koji animira svojstva transformacija
4. <*animateColor*>, koji omogućuje mijenjanje boje u jedinicama vremena
5. <*set*>, koji mijenja vrijednosti atributa koji nisu numerički, poput atributa *visibility*

Element *set* koristi se za rukovanje upečatljivih vizualnih promjena između dva atributivna stanja. Ostala četiri elementa u pravilu rade sličnu stvar, no tranzicije između dva ili više stanja su postepena, a ne iznenadna kao što je slučaj sa *set* elementom.

Kao dodatak navedenim elemenata za animacije, SVG uključuje ekstenzije i atribute koje proširuju mogućnosti navedenim elementima. Među njima su

1. *path* (atribut) – Omogućuje specifikaciju bilo kojeg svojstva SVG putanje unutar *animateMotion* elementa.
2. <*mpath*> – Element koji je usko povezan s *animateMotion* elementom, a definira putanju po kojoj će se kretati objekt. Postavlja se unutar *animateMotion* elementa, prije zatvaranja njegovog pripadajućeg taga.
3. *keypoints* (atribut) – Koristi se kao atributi unutar *animateMotion* elementa, a kontrolira brzinu kretanja objekta po putanji.
4. *rotate* (attribute) – Navedeni atribut koristi se u sklopu *animateMotion* elementa te kontrolira automatsku rotaciju objekta ovisno o tangentnom vektoru putanje po kojoj se objekt kreće.

SVG objekte moguće je animirati i putem CSS-a, no postoje određena svojstva i atributi unutar SVG-a koje CSS ne može obraditi i animirati. Primjer takve iznimke je path element, točnije njegov atribut *d* (*data*), pomoću kojeg se definiraju točke putanje. Praznine koje se ne mogu animirati putem CSS-a moguće je zamijeniti *JavaScript*-om ili pak SMIL-om, kao što je slučaj u ovom radu.

SMIL ima vrlo veliku prednost nad JavaScript animacijama, a ta je što JavaScript animacije ne rade u slučaju kada je SVG smješten unutar *img* ili *background-image* atributa u CSS-u. SVG animacije funkcioniraju u oba slučaja, ukoliko internetski preglednik nudi potrebnu podršku za SVG.

Što se same podrške internetskih preglednika tiče, većina preglednika podržava većinu svojstava SMIL animacija. Iznimke su Internet Explorer te Opera Mini.

Elementi za animacije postavljaju se ili unutar elementa koji se želi animirati ili zasebno. Ukoliko se animacija piše odvojeno od objekta na koji se odnosi, tada je potrebno definirati objekt ili grupu objekata koja će se animirati, a to se radi pomoću *xlink:href* atributa. Prethodno je potrebno dati *id* atribut, to jest ime, tom objektu, kako bi ga kasnije mogli pozvati unutar koda animacije.

Drugi najvažniji atribut koji se primijenjuje kod svih vrsta animacija u SVG-u je *attributeName*. Koristi se za određivanje imena onog atributa objekta koji će se animirati. Ukoliko je u pitanju pravokutnik (<*rect*>), moguće je animirati na primjer vrijednost *x* atributa. U tom će se slučaju pravokutnik pomicati po x osi.

AttributeName može mijenjati samo jedan atribut, stoga ukoliko je na primjer potrebno izraditi pravokutnik koji se istodobno pomiče po x osi i mijenja širinu, tada je potrebno odvojeno definirati dvije animacije, svaku za odgovarajući atribut.

Kako bi se promijenile vrijednosti unutar vremena animacije, koriste se *from*, *to* i *dur* atributi. Atribut *from* označava početno stanje atributivne vrijednosti, dok *to* označava završno stanje istog. Atribut *dur*, skraćeno od riječi trajanje (eng. *duration*), određuje trajanje vremenskog perioda u kojem će se odvijati animacija. Također se koristi i *begin* atribut, koji označava u kojem će trenutku animacija početi. Vrijednost *begin* atributa može biti *click*, što znači da animacija počinje klikom miša, zatim 0s, što znači da počinje odmah, ili pak s određenom odgodom, poput 5s, to jest 5 sekundi. Također se može koristiti kombinacija ove dvije metode, na primjer *begin="click+1s".*

Atribut fill definira poziciju elementa nakon što se odvije animacija. Ukoliko je definirana vrijednost *freeze*, tada će objekt ostati na zadnjoj poziciji koja je definirana animacijom, a ako je zadana vrijednost *remove*, tada animacija

nakon svog završetka više nema utjecaja na svojstva objekta koji je prethodno animiran.

Ukoliko se traži ponavljanje animacije više od jednog puta, koristi se *repeatCount* atribut. Moguće vrijednosti su *indefinite*, što znači da će se animacija odvijati bezbroj puta, te neka brojčana vrijednost. Za razliku od CSS-a, SMIL ne dopušta reverzno svojstvo animacije. Unutar SMIL-a, to je moguće samo korištenjem *JavaScript*-a.

Kada se specificiraju vrijednosti animacije po *frameovima*, koriste se *keyTimes* i *values* atributi. Atribut *keyTimes* definira postotak vremena u kojem će se primijeniti vrijednosti iz values atributa. Vrijednosti mogu biti između 0 i 1. U *values* atributu ispisuju se vrijednosti koje *keyTimes* atribut poziva prilikom trajanja animacije. [7]

Hod animacije definira se *calcMode* atributom. Postoje četiri različite vrijednosti *calcMode* atributa:

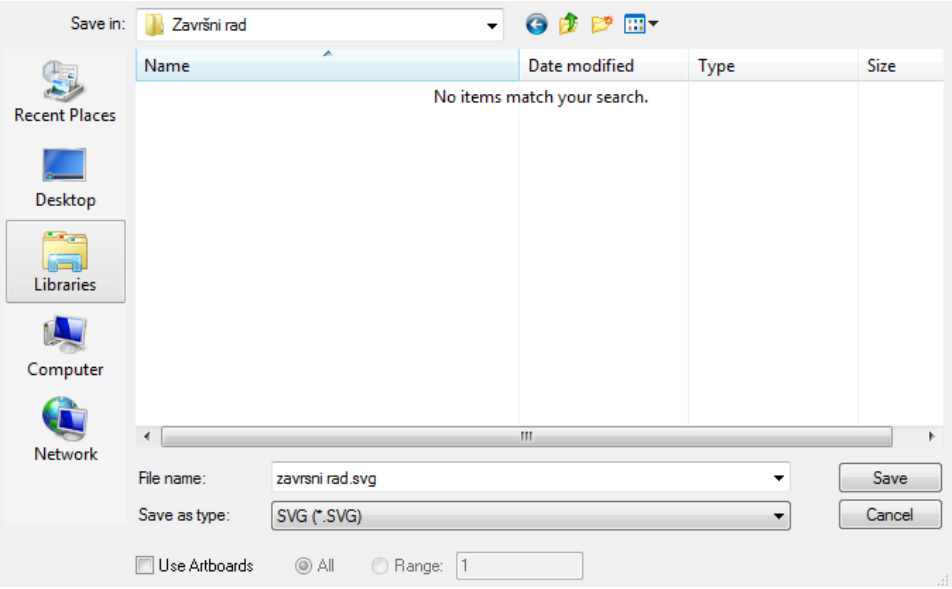
- *linear*
- *discrete*
- *paced*
- *spline*

Ukoliko *calcMode* nije definiran, tada je hod animacije linearan (*linear*). Vrijednost *paced* je vrlo sličan *linear* vrijednosti, osim što ne koristi vrijednosti definirane unutar *keyTimes* atributa. Vrijednost *discrete* omogućuje skok između jedne vrijednosti na druge bez interpolacije. Četvrta vrijednost, spline, mijenja vrijednosti u vremenu pomoću kubične Bezierove krivulje. Intervali animacije su definirani pomoću keyTimes atributa, a točke kubične Bezierove krivulje pomoću keySpline atributa. [8]

2.8. Korištenje Adobe Illustratora za stvaranje ilustracija

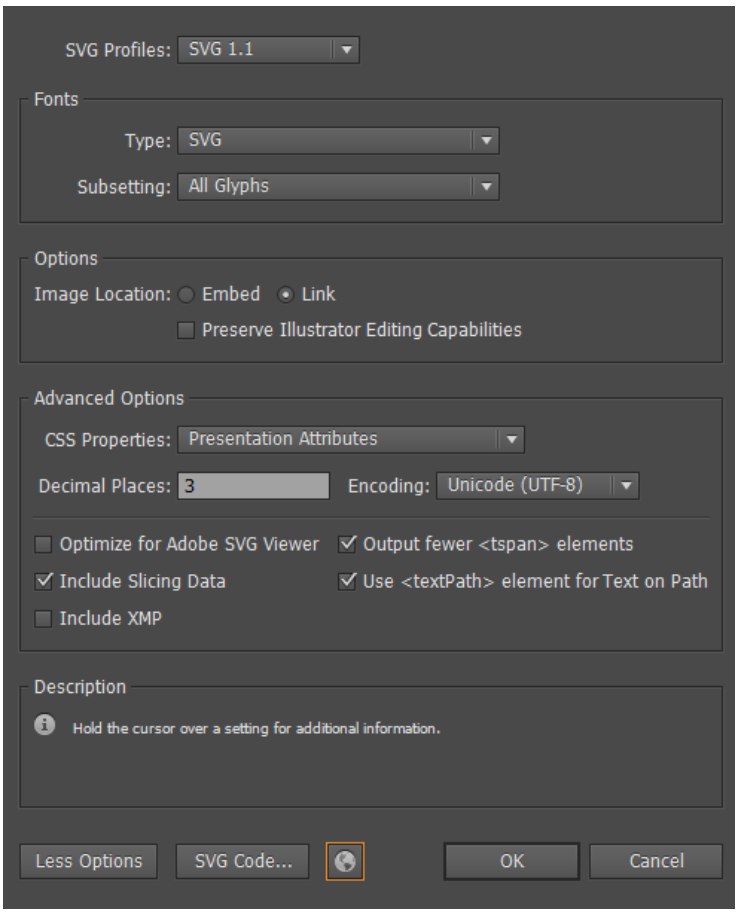
Korištenjem Adobe Illustratora te sličnih programa za crtanje vektorske grafike koji podržavaju mogućnost spremanja kao SVG dokument olakšano je i ubrzano crtanje kompleksnih oblika te stvaranja uzoraka i gradijenata. Adobe Illustrator prilikom spremanja SVG dokumenta sam generira kod, koji se naknadno može otvoriti i uređivati u Adobe Illustratoru, ali se mogu i uređivati dijelovi koda pomoću bilo kojeg uređivača teksta.

Ukoliko korisnik želi spremiti vektorsku grafiku stvorenu u Adobe Illustratoru i želi ju spremiti kao SVG dokument, potrebno je izabrati *File* > *Save As*. Tada se otvara novi prozor, pri čemu je potrebno odrediti ime dokumenta te format dokumenta, u ovom slučaju SVG(*.SVG), kao što je prikazano na Slici 13. Ispod iskočnog izbornika za format dokumenta, nalazi se *Use Artboards* opcija, koji omogućava spremanje dijelova vektorske grafike kao zasebnih dokumenata. [9]



Slika 13. Spremanje SVG datoteke

Kada se sve navedeno odradi, potrebno je kliknuti Save, nakon čega se otvara novi prozor s SVG opcijama, prikazan na Slici 14.



Slika 14. Svojstva SVG datoteke

Najbolje je odabrati sljedeće opcije za *web*:

- *Profile: SVG 1.1*
- *Fonts: Type SVG – Only glyphs used*
- *Image location: Link*
- *CSS Properties: Style Elements*
- *Decimal Places: 1*
- *Output fewer <tspan> elements*
- *Use <textPath> element for Text on Path*

Prilikom spremanja dokumenta kao SVG, imena *layera* i objekata se spremaju nešto drugačije, nego što su definirani u Adobe Illustratoru. Ukoliko ime ima razmak, tada će se u SVG kodu, kao vrijednost *id* atributa, razmak zamijeniti sa znakom „_“.

Ukoliko grafika koja se sprema u SVG sadrži kompleksne oblike, tada će dobiveni kod biti vrlo dugačak, a sama veličina dokumenta vrlo velika. Stvorit će se nekoliko stotina linija koda. Kako bi se to spriječilo, potrebno je koristiti

odgovarajuće oblike. Tako se primjerice umjesto niza pravokutnika koji se nalaze jedan do drugog, može se napraviti veliki pravokutnik, unutar čega se nacrtaju linije. Osim što će datoteka biti manja, također se smanjuje mogućnost pogrešaka prilikom prikaza dokumenta u različitim SVG preglednicima, s obzirom da u njima može doći do pomaka od 0.0001 piksela, što znatno može utjecati na prikaz.

U slučaju kada se izvozi SVG iz Adobe Illustratora, kod postavljanja decimalne vrijednosti najbolje je upisati numeričku vrijednost „1“, no samo ako ne postoji mnogo krivulja u vektorskoj grafici.

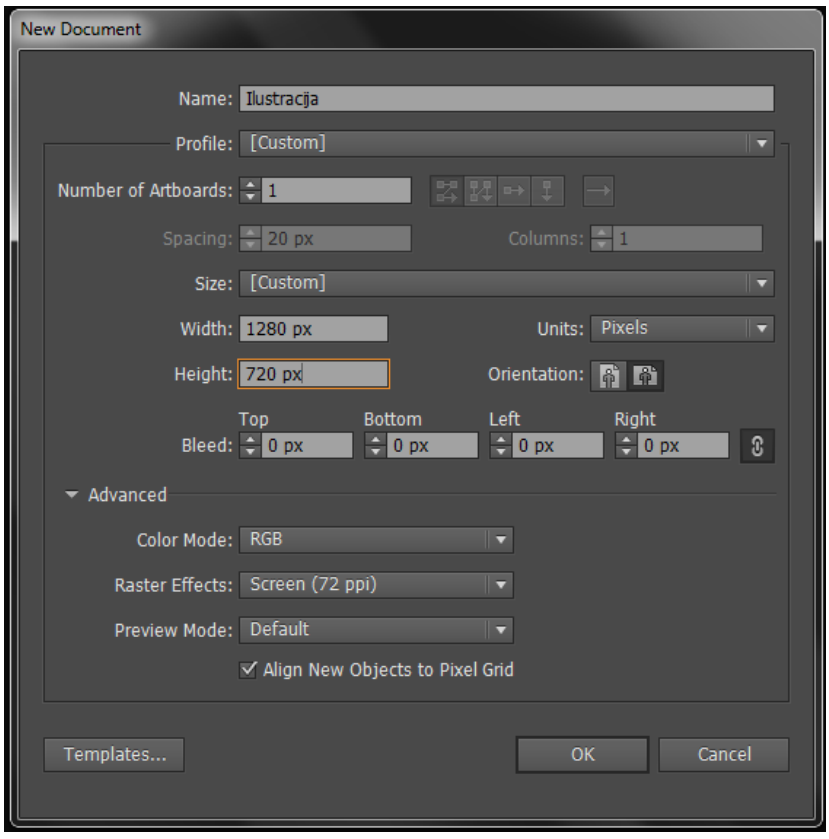
Najnovija verzija Adobe Illustratora, Illustrator CC, omogućava stvaranje responzivnog SVG koda, što znači da ukoliko se smanjuje prozor internetske stranice, tada će se i SVG grafika smanjivati recipročno sa smanjivanjem prozora. [10]

3. EKSPERIMENTALNI DIO

Vektorska grafika iz ovog rada sastoji se od šest osnovnih grupa elemenata: pozadine, eksplozije, Pokemona, lopte, Pokemonove sjene te sjene lopte. Prije samog crtanja ilustracije u Adobe Illustratoru, potrebno je osmisliti koncept animacije, kako bi se elementi na prikladan način grupirali. Iste je potrebno i imenovati radi lakšeg snalaženja pri dodavanju animacija na njih.

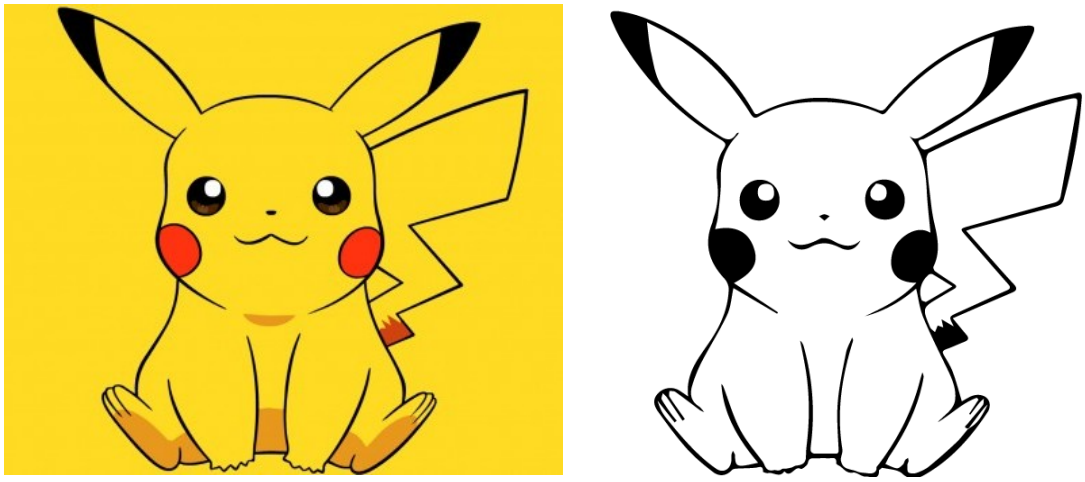
3.1. Stvaranje ilustracije u Adobe Illustratoru

Nakon otvaranja Adobe Illustratora, potrebno je stvoriti novu datoteku klikom na File/New..., nakon čega se otvori novi prozor sa osnovnim opcijama nove datoteke. Ilustracija iz ovog rada je široka 1280 piksela i visoka 720 piksela, kao što je prikazano na Slici 15.



Slika 15. Opcije novog dokumenta

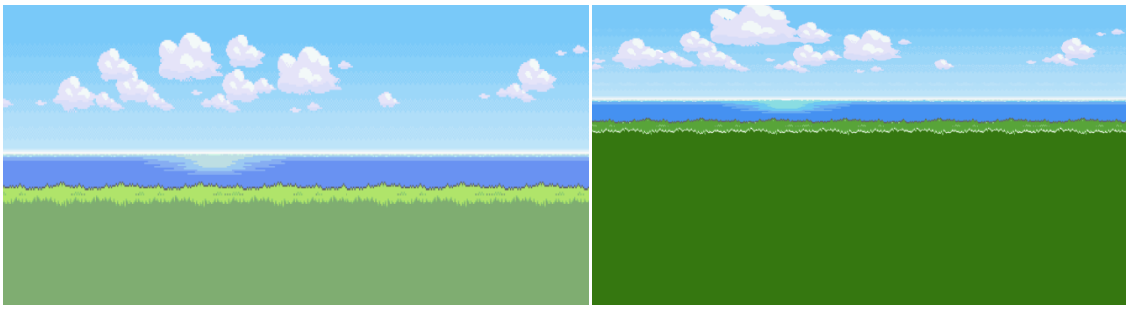
Nakon toga otvara se prazni dokument i može se početi s crtanjem. Prva grupa elemenata crtana je pomoću referentne te su obrisi stvoreni pomoću opcije *Image Trace* u crno-bijelom načinu rada (Slika 16).



Slika 16. Referentna slika (lijevo) i obrisi stvoreni opcijom *Image Trace* (desno)

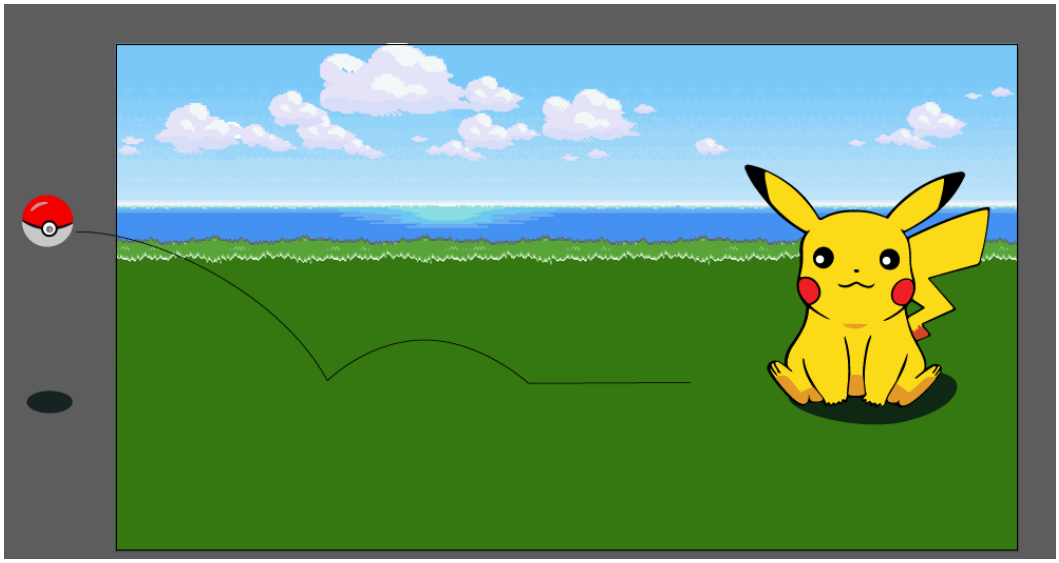
Obzirom da određeni detalji nisu dovoljno dobro automatski stvoreni, potrebno ih je ručno nacrtati i popraviti te je ilustraciju potrebno i obojati, što je naknadno učenjeno opcijom *Live Paint Bucket*. Grupacija elemenata lopte crtana je ručno pomoću *Pen Toola*, *Ellipse Toola* te *Pathfindera*, nakon čega su elementi pobojani *Live Paint Bucket* opcijom ili pak primjenom gradijenta. Sjene su stvorene crtanjem elipse te obojane u željenu boju.

Prilikom stvaranja pozadine također je korištena opcija Image Trace, no ovoga puta u boji. Određeni dijelovi su uvećani te im je promijenjena ispuna. Usporedba refentne slike i pozadine ove SVG grafike nalazi se na Slici 17.



Slika 17. Referentna slika (lijevo) i krajnji rezultat stvoren opcijom *Image Trace* (desno)

Kada su svi navedeni elementi vidljivi, ilustracija izgleda ovako:



Slika 18. Gotova ilustracija

Također je važno napomenuti da se na Slici 18 vidi i putanja po kojoj će se kretati lopta, no kasnije će se ista ukloniti. Vrlo je važno navedenu putanju dodati kako bi prilikom spremanja kao SVG dokument bile pohranjene i koordinate točaka putanje za *animateMotion* element.

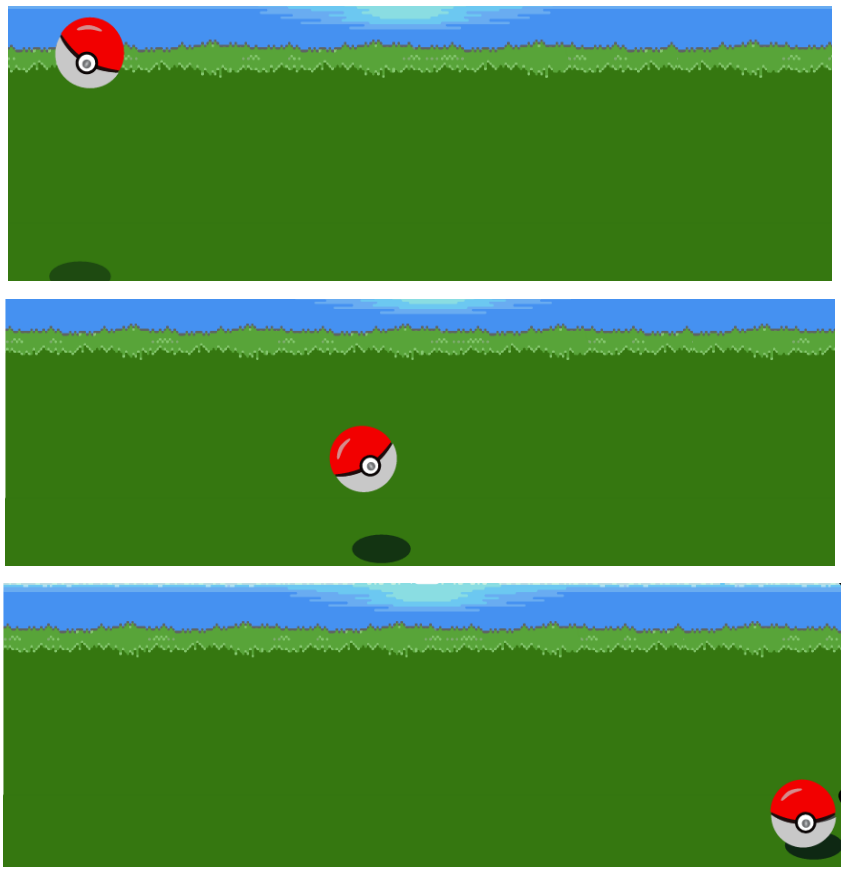
Na kraju je potrebno spremiti dokument u .svg formatu, kao što je prethodno opisano u teorijskom dijelu rada.

3.2. Dodavanje animacija na elemente ilustracije

Nakon spremanja SVG dokumenta, potrebno je otvoriti datoteku u uređivaču teksta. U ovom radu korišten je Notepad++. Adobe Illustrator generirao je SVG kod koji je potrebno smjestiti u <*body*> tag HTML dokumenta, nakon čega se mogu početi animirati dijelovi ilustracije.

Prva grupa elemenata koja je animirana je grupacija elemenata imenovana „Pokelopta“. Unutar nje nalaze se dvije manje grupe elemenata: gornji dio objekta te donji dio objekta. Ovakva podjela vrlo je bitna za kasniju primjenu rotacije samo na gornji dio grupe Pokelopta.

Na početku je potrebno prikazati horizontalni pad, točnije translaciju lopte po putanji, što je moguće izvesti umetanjem *animateMotion* taga unutar navedene grupe elemenata. Dijelovi ove animacije prikazane su na Slici 19.



Slika 19. Prikaz *animateMotion* elementa lopte

```
<animateMotion
      path="M-57.65,217.345 c110.149-
      1.645,300.149,104.526,372.149,264.44c0,0,60-
      72.306,144-72.306s154,76.995,154,76.9951240.053-1.386"
      additive="sum"
      begin="0s"
      dur="3s"
      repeatCount="1"
      rotate="auto"
      fill="freeze"
      />
```

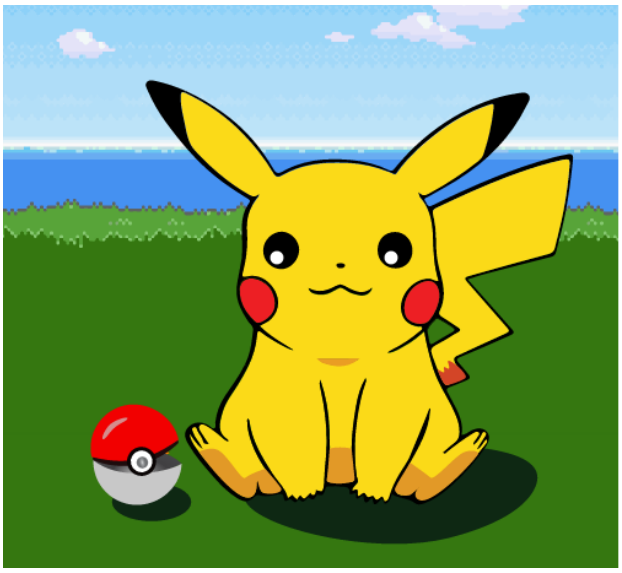
Zahvaljujući *additive* atributu i pripadajućoj vrijednostin *sum* omogućeno je da se više od jedne animacije primijenjuje na isti tag. Ukoliko se navedeni atribut ne dodaje, tada će raditi isključivo samo jedna animacija. Vrijednosti atributa path kopirani su iz <*path*> taga koji se nalazio u izvornog SVG dokumentu te je <*path*> tag obrisan s obzirom da nije potreban. Atribut rotate određuje rotaciju

animiranog objekta u usporedbi s tangentom krivulje po kojoj se objekt translatira.

Sukladno s animacijom lopte, animirana je i sjena iste, točnije primijenjena je translacija po osi. Također se mijenja opacitet sjene u vremenu.

```
<animate
     attributeName="cx"
     attributeType="XML"
     dur="3s"
     values="0; 250; 540; 900"
     keyTimes="0; 0.32; 0.66; 1"
     repeatCount="1"
     fill="freeze"/>
<animate
     attributeName="opacity"
     dur="3s"
     values="0.2 ; 0.8 ; 0.5 ; 0.8; 0.8"
     keyTimes="0 ; 0.4 ; 0.66 ; 0.8; 1"
     repeatCount="1"
     fill="freeze"/>
```

Nakon prestanka prve animacije, gornji dio grupe počinje se rotirati oko centra koordinatnog sustava pod kutem od 0 o -20°, a nakon trenutaka kasnije i u suprotnom smjeru. Ove animacije imaju tag *animateTransform*, čija je vrijednost *type* atributa *rotate*. Rotacija je prikazana na slici 20.



Slika 20. Rotacija gornjeg dijela lopte

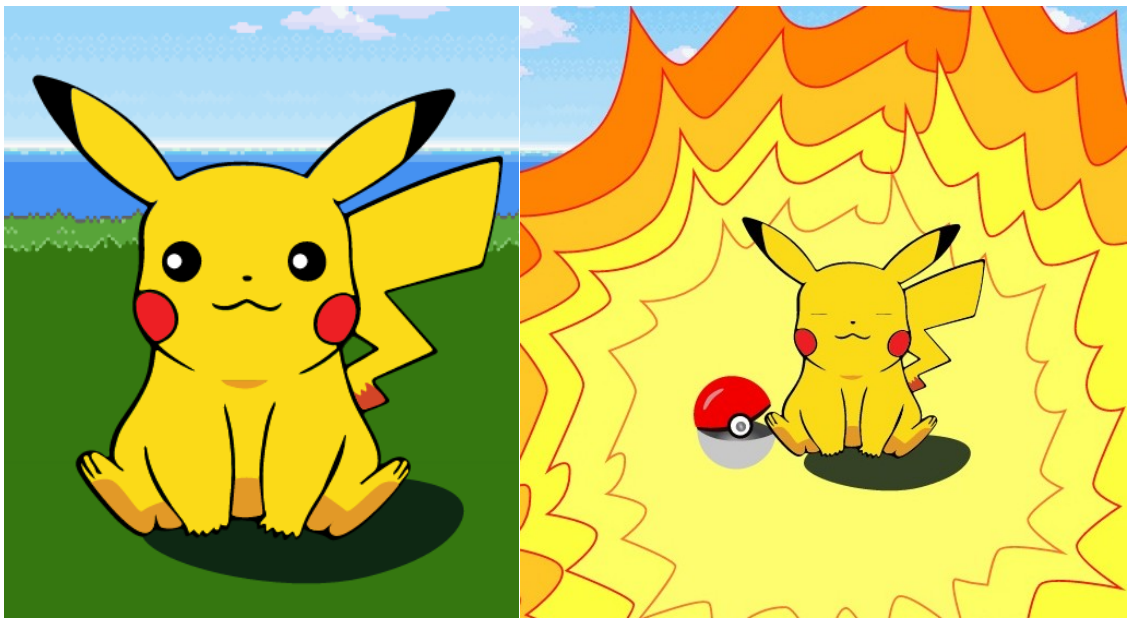
```
<animateTransform
      attributeName="transform"
      type="rotate"
      additive="sum"
      begin="3s"
      dur="3s"
      from="0 0 0"
      to="-20 0 0"
      fill="freeze"/>

<animateTransform
      attributeName="transform"
      type="rotate"
      additive="sum"
      begin="8s"
      dur="3s
      from="0 0 0"
      to="20 0 0"
      fill="freeze"/>
```

Na samom kraju dodana je i animacija, točnije rotacija, na cijelu „Pokelopta“ grupu. Vrijednosti rotacije definirane unutar *keyTimes* atributa te se iste manifestiraju u periodima definiranima u *values* atributu. Ova animacija vidljiva je na Slici 21.

Slika 21. Rotacija lopte

```
<animateTransform
      attributeName="transform"
      type="rotate"
      additive="sum"
      begin="8s"
      dur="2s"
      values="10 50 100;-10 50 100;10 50 100;-10 50 100; 10
      50 100; -10 50 100; 10 50 100; -10 50 100; 10 50 100;-
      10 50 100;10 50 100"
      keyTimes="0;0.1;0.2;0.3;0.4;0.5;0.5;0.7;0.8;0.9;1"
      fill="freeze"/>
```

Nakon što je animirana grupa elemenata „Pokelopta“, bilo je potrebno animirati grupu elemenata „Pikachu“. Smanjuje se, oči se smanjuju po *y* osi te se zjenice rotiraju istovremeno kada se „Pokelopta“ kreće po putanji. Smanjivanje cijele grupe te smanjivanje očiju po y osi prikazane su na Slici 22.



Slika 22. Prikaz animacija na grupi elemenata Pikachu

Primjer rotacije zjenice izgleda ovako:

```
<animateTransform
      attributeName="transform"
      type="rotate"
      additive="sum"
      begin="0s"
      dur="3s"
      keyTimes="0;0.35;0.65;0.75;1"
      values="0 0 0;-60 0 0;-30 0 0;-80 0 0;-90 0 0"
      fill="freeze"/>
```

Treptanje je realizirano smanjivanjem grupe elemenata koji ujedinjuju šarenice i zjenice, isključivo po *y* osi.

```
<animateTransform
      attributeType="XML"
      attributeName="transform"
      type="scale"
      additive="sum"
      values="1 1;1 1;1 0.02;1 1;1 1;1 0.02;1 1;1 1;1 0.02;1
      0.02"
```

```
     keyTimes="0;0.19;0.2;0.21;0.59;0.6;0.61;0.97;0.99;1"
     begin="1s"
     dur="5s"
     fill="freeze"/>
```

Prilikom smanjivanja cijele grupacije elemenata „Pikachu“, istovremeno se odvija smanjivanje elementa „SjenaPikachu“.

```
<animateTransform
     attributeType="XML"
     attributeName="transform"
     type="scale"
     additive="sum"
     from="1"
     to="0"
     begin="6s"
     dur="0.5s"
     fill="freeze"/>
```

U grupu objekata „Eksplozija“ dodan je *set* za opacitet te *scale* od vrijednosti 0 do vrijednosti 1.

```
<set
     attributeType="CSS"
     attributeName="visibility"
     additive="sum"
     to="visible"
     begin="5s"
     dur="5s"
     fill="remove"/>

<animateTransform
     attributeType="XML"
     attributeName="transform"
     type="scale"
     additive="sum"
```

```
keyTimes="0;0.2;1"
values="0;1;0"
begin="5s"
dur="5s"
fill="remove"/>
```

4. ZAKLJUČAK

U ovom radu prikazan je i opisan primjer vektorske grafike na *webu* koja je potom animirana isključivo koristeći SVG animacije. Postupak izrade ilustracije znatno je olakšan mogućnošću spremanja datoteke u Adobe Illustratoru u .svg formatu. Na ovaj se način brzo i jednostavno mogu stvoriti vizualno privlačne vektorske grafike.

Iako se SVG animacije sve češće zamijenjuju CSS animacijama te se na iste primjenjuje *JavaScript*, postoje određena svojstva SVG animacija koje CSS animacije ne podržavaju, poput *path* svojstava. Veliki nedostatak SVG animacija je što je podrška internetskih preglednika za mobilne uređaje lošija od podrške CSS animacija.

Najveći problem pri izradi SVG animacija je primjena više od jedne transformacije na neku grupu elemenata, što je postupak koji može postati vrlo mukotrpan. Transformacije elemenata mogu vrlo lako zbuniti dizajnera. Najveći problem nastaje kod *scale* transformacije, gdje se ne može na jednostavan način upisati koordinate centra *scalea*.

Unatoč navedenim nedostacima, SVG animacije ne trebaju biti potpuno zanemarene i ostavljene po strani u korist konkurentskih načina stvaranja animacija na internetskim stranicama, pošto omogućuju brzo i jednostavno kreiranje jednako kompleksnih i vizualno zanimljivih animacija.

5. LITERATURA

1. Eisenberg J. D. (2002.) *SVG Essentials*, O'Reilly & Associates, Sebastopol
2. Laaker M. (2002.) *SAMS Teach Yourself SVG in 24 Hours*, Sams Publishing, Indianapolis
3. https://www.w3.org/TR/SVGTiny12/intro.html#SVGTiny12 (27. srpnja 2016.)
4. https://www.w3.org/TR/REC-smil/smil-animation.html (27. srpnja 2016.)
5. https://www.w3.org/TR/SVG/pservers.html (17. srpnja 2016.)
6. https://developer.mozilla.org/en-US/docs/Web/SVG/Tutorial (17. srpnja 2016.)
7. https://css-tricks.com/guide-svg-animations-smil/ (16. kolovoza 2016.)
8. Dawber D. (2013.). *Learning Raphaël JS Vector Graphics*, Packt Publishing Ltd., Birmingham
9. http://creativedroplets.com/export-svg-for-the-web-with-illustrator-cc/ (16. kolovoza 2016.)
10. https://helpx.adobe.com/illustrator/how-to/responsive-svg-files-in-illustrator.html (16. kolovoza 2016.)