Fakultet elektrotehnike, računarstva i informacijskih tehnologija Osijek

Vizualizacija podataka

Seminarski rad
Vizualizacija statistika Pokemona

Ivan Sertić

Osijek, 2020.

Contents

1. Uvod

U ovom projektnom zadataku napravljena je web aplikacija koja grafički prikazuje statistike *Pokemona* prve generacije. Vizualizacija ima mogućnost promjene prikaza statistika *Pokemona* na pritiskom tipke miša na sliku željenog *Pokemona*. Svaki *Pokemon* ima svoje statistike koje se prikazuju pomoću *Radar Charta.*

Pritiskom tipke miša na željenu sliku prikazuje se *Radar Chart* koji vizualno prikazuje statistike odabranog *Pokemona. Radar Chart* također nije statičen već se prikazuje u redu u kojem se nalazi i odabrana slika kako korisnik nebi morao za svaki prikaz podizati stranicu.

Radar Chart prikazuje HP, Attack, Defense, Sp. Attc, Sp. Def i Speed pojedinog *Pokemona.*

Web aplikacija je izrađena pomoću JavaScripta[1] i D3.js[2] biblioteke pomoću koje su generirani HTML i SVG elemnti. Bootstrap[3] je korišten kako bi slike bile prikazane u redovima i stupcima.

2. Skup podataka

U ovom poglavlju bit će opisan skup brojčanih podataka koji je korišten za svakog *Pokemona* te skup slika koji je povezan sa izgledom svakog od njih.

2.1. Skup brojčanih statistika Pokemona

Ovaj skup sadržavao je sve statistike svih Pokemona te detaljan opis istih. Kako je odlučeno da će projekt prikazivati samo prvu generaciju bilo je potrebno filtrirati iz skupa samo Pokemone prve generacije. Nakon filtriranja dobio se sljedeći skup podataka.

Number	Name	Type 1	Type 2	Total	HP	Attack	Defense	Sp. Atk	Sp. Def	Speed	Generation	Legendary
1	Bulbasaur	Grass	Poison	318	45	49	49	65	65	45	1	0
2	Ivysaur	Grass	Poison	405	60	62	63	80	80	60	1	0
3	Venusaur	Grass	Poison	525	80	82	83	100	100	80	1	0
4	Charmander	Fire		309	39	52	43	60	50	65	1	0
5	Charmeleon	Fire		405	58	64	58	80	65	80	1	0
6	Charizard	Fire	Flying	534	78	84	78	109	85	100	1	0
7	Squirtle	Water		314	44	48	65	50	64	43	1	0
8	Wartortle	Water		405	59	63	80	65	80	58	1	0
9	Blastoise	Water		530	79	83	100	85	105	78	1	0

10	Caterpie	Bug		195	45	30	35	20	20	45	1	0
11	Metapod	Bug		205	50	20	55	25	25	30	1	0
12	Butterfree	Bug	Flying	395	60	45	50	90	80	70	1	0
13	Weedle	Bug	Poison	195	40	35	30	20	20	50	1	0
14	Kakuna	Bug	Poison	205	45	25	50	25	25	35	1	0
15	Beedrill	Bug	Poison	395	65	90	40	45	80	75	1	0
16	Pidgey	Normal	Flying	251	40	45	40	35	35	56	1	0
17	Pidgeotto	Normal	Flying	349	63	60	55	50	50	71	1	0
18	Pidgeot	Normal	Flying	479	83	80	75	70	70	101	1	0
19	Rattata	Normal		253	30	56	35	25	35	72	1	0
20	Raticate	Normal		413	55	81	60	50	70	97	1	0
21	Spearow	Normal	Flying	262	40	60	30	31	31	70	1	0
22	Fearow	Normal	Flying	442	65	90	65	61	61	100	1	0
23	Ekans	Poison		288	35	60	44	40	54	55	1	0
24	Arbok	Poison		438	60	85	69	65	79	80	1	0
25	Pikachu	Electric		320	35	55	40	50	50	90	1	0

26	Raichu	Electric		485	60	90	55	90	80	110	1	0
27	Sandshrew	Ground		300	50	75	85	20	30	40	1	0
28	Sandslash	Ground		450	75	100	110	45	55	65	1	0
29	Nidoran♀	Poison		275	55	47	52	40	40	41	1	0

Tablica 1: Prikaz dijela dobivenog skupa podataka koji je korišten

Nakon što je dobijen potreban skup podataka podatke je bilo potrebno pretvoriti u JSON format kako bi njihov prikaz i njihovo korištenje bilo moguće unutar JavaScripta i D3.js biblioteke. Pretvorba CSV datoteke u JSON datoteku izvršena je pomoću csv2json[4].

```
var pokedex = [{
    "Number": 1,
    "Name": "Bulbasaur",
    "Type 1": "Grass",
    "Type 2": "Poison",
    "Total": 318,
    "HP": 45,
    "Attack": 49,
    "Defense": 49,
    "Sp. Atk": 65,
    "Sp. Def": 65,
    "Speed": 45,
    "Generation": 1,
    "Legendary": false
},
{
    "Number": 2,
    "Name": "Ivysaur",
    "Type 1": "Grass",
    "Type 2": "Poison",
    "Total": 405,
    "HP": 60,
    "Attack": 62,
    "Defense": 63,
    "Sp. Atk": 80,
    "Sp. Def": 80,
    "Speed": 60,
    "Generation": 1,
    "Legendary": false
},
{
    "Number": 3,
    "Name": "Venusaur",
    "Type 1": "Grass",
    "Type 2": "Poison",
    "Total": 525,
    "HP": 80,
    "Attack": 82,
    "Defense": 83,
    "Sp. Atk": 100,
    "Sp. Def": 100,
    "Speed": 80,
    "Generation": 1,
    "Legendary": false
},
```

Isječak koda 1: Prikaz polja koji sadrži JSON objekte unutar JS datoteke

2.2 Skup slika

Skup slika za prikaz na web stranici korišten je kao skup podataka. Podatci, odnosno slike, spremljeni su u zaseban direktorij unutar projekte te su iz nejga pozivani tj. prikazani.

Slika 1: Prikaz slika unutar direktorihja

3. Implementacija

U ovom poglavlju bit će opisani načini implementacije web aplikacije.

3.1 Prilagodba JSON datoteke

Kako bi bilo lakše korištenje dobijene JSON datoteke stvorena je nova .JS datoteka u kojoj je deklarirana varijabla koja sadrži polje JSON objekata. Nakon toga je bilo potrebono pozvati JS datoteku unutar glavne index.html datoteke.

```
<link rel="stylesheet" href="https://maxcdn.bootstrapcdn.com/bootstrap/3.4.1/css/bootstrap.min.css">
<script src="https://d3js.org/d3.v5.min.js"></script>
<script src="pokedex.js"></script>
```

Isječak koda 2: Prikaz pozivanja skripti unutar index.html datoteke

Ovakav način je odabran radi lakšeg korištenja JSON podataka te kako bi podatci bili dostupni unutar globalnog dijela skripte.

3.2. Prikaz slika

Kako bi slike bile prikazane korištena je D3.js biblioteka. Pomoću nje stvoreni su HTML „div“ elementi kojima je dodjeljena Bootstrap klasa kako bi podatci bili prikazani na način da se grupiraju u tri stupca. Zato je bilo potrebno staviti izvor slike kako bi stranica znala koju sliku gdje treba prikazati. Pošto su JSON objekti unutar polja sortirani redosljedom kojim se *Pokemon* pojavljuje u *Pokedexu* bilo je potrebno samo iterirati po polju pomoću for petlje. Iduće je bilo ptrebno dodati „src“ za „image“ element. Slike Pokemona nazvane su kao ime pojedinog pokemona stoga je samo bilo potrebno dohvatiti atribut „Name“ pojedinog JSON objekta, pretvoriti ga u mala slova i dodati mu nastavak „.png“. Nakon toga slike su prikazane na stranici.

```
44          function showPictures(data1, data2, data3) {
45              let data1WFeatures = {};
46              spiderChardFetures.forEach(f=>{
47                  data1WFeatures[f] = data1[f]
48              });
49
50              let divContainer = d3.select("body")
51                  .append("div")
52                  .attr("class", "container");
53              let divRow = divContainer.insert("div")
54                  .attr("class", "row col-md-12");
55              let image1 = divRow.insert("div")
56                  .attr("class", "col-md-4");
57              image1 = image1.insert("div").attr("align", "center")
58              let imSvg = image1.insert("svg").attr("width", "120").attr("height", "120");
59              imSvg.insert("image").attr('xlink:href', "/pokemonImages/" + data1.Name.toLowerCase() + ".png");
60              image1.insert("br");
61              image1.insert("text").text(data1.Name);
```

Isječak koda 3: Primjer stvaranja elemenata za prikaz slika

3.3. Postavljanje funkcionalnosti klika

Kako bi svaka slika imala funkcionalnost prikaza Radar Charta na klik, bilo je potrebno na svaku sliku dodati funkciju koja će se za to pobrinuti.

```
63                  imSvg.on("click", function () {
64                      plotTheData(data1WFeatures,data1.Name);
65                      return info
66                          .style('visibility', 'visible')
67                          .style('top', (event.pageY-60) + 'px')
68                          .style('left', 100 + 'px')
69                  })
```

Isječak koda 4: Primjer postavljanja funkcije za rukovođenje klika

3.4. Stvaranje Radar Charta

Kako bi se prikazali podatci odabran je *Radar Chart* jer je najpregledniji i najzastupljeniji unutar igara igranja uloga(*RPG*). Prije samog prelaska na izradu ovog dijagrama napravljen je „div“ element koji u sebi sadržava „svg“ element. Kako taj element nebi „smetao“ na stranici kada na niti jednu sliku još nije kliknuto taj „div“ lement je sakriven dok se ne desi prvi klik.

```
20          let info = d3
21              .select('body')
22              .append('div')
23              .style('position', 'absolute')
24              .style('z-index', '10')
25              .style('visibility', 'hidden');
26
27          let svgInDiv = info.insert("svg")
28              .attr("width", 300)
29              .attr("height", 300);
30
```

Isječak koda 5: Prikaz koda koji stvara početni svg element unutar div elementa

Nakon što je napravljen „svg“ element unutar njega je bilo potrebno dodati kružnice koje će prikazivati vrijednosti statistika te je bilo potrebno odabrati skalu i kako će se ta skala prikazati.

```
let marks = [50, 100, 150, 200, 250];

marks.forEach(m =>
    svgInfo.insert("circle")
        .attr("cx", 150)
        .attr("cy", 150)
        .attr("fill", "none")
        .attr("stroke", "gray")
        .attr("r", radialScale(m))
);

marks.forEach(m => {
    svgInfo.append("text")
        .attr("x", 155)
        .attr("y", 150 - radialScale(m))
        .text((m).toString())
})
```

Isječak koda 6: Prikaz koda za stvaranje kružnica

```
17          let radialScale = d3.scaleLinear()
18                  .domain([0, 250])
19                  .range([0, 140]);
```

Isječak koda 7: Prikaz koda stvaranja skale

Nakon što su kružnice i skala bili nacrtani potrebno je bilo stvoriti osi koje će prikazivati imena pojedinih atributa.

```
170         for (var i = 0; i < spiderChardFetures.length; i++) {
171             let name = spiderChardFetures[i];
172             let angle = (Math.PI / 2) + (2 * Math.PI * i / spiderChardFetures.length);
173 
174             let line_coordinate = angleToCoordinates(angle, 250, radialScale);
175             let label_coordinate = angleToCoordinates(angle, 170, radialScale);
176 
177             svgInfo.append("line")
178                 .attr("x1", 150)
179                 .attr("y1", 150)
180                 .attr("x2", line_coordinate.x)
181                 .attr("y2", line_coordinate.y)
182                 .attr("stroke", "black");
183 
184             svgInfo.append("text")
185                 .attr("x", label_coordinate.x)
186                 .attr("y", label_coordinate.y)
187                 .attr("fill","red")
188                 .text(name);
189         }
```

Isječak koda 8: Prikaz koda za stvaranje osi i stavljanje pojedinih atributa

Sada kada je stvoren izgled *Radar Charta* potrebno je na njega staviti „path“ koji će prikazivati podatke. Kako bi bilo moguće dinamičko prikazivanje podataka stavljeni su podatci sa kordinatama x je 0 i y je 0. Ovakav izgled *Radar Charta* nije vidljiv korisniku jer je takav graf u početku sakriven.

```
191             let initial = [
192                 {
193                     x: 0,
194                     y: 0
195                 },
196                 {
197                     x: 0,
198                     y: 0
199                 },
200                 {
201                     x: 0,
202                     y: 0
203                 },
204                 {
205                     x: 0,
206                     y: 0
207                 },
208                 {
209                     x: 0,
210                     y: 0
211                 },
212                 {
213                     x: 0,
214                     y: 0
215                 },
216                 {
217                     x: 0,
218                     y: 0
219                 }
220             ]
221 
222             let line = d3.line()
223                 .x(d => d.x)
224                 .y(d => d.y);
225 
226             svgInDiv.append("text")
227                 .attr("id","pokeName")
228                 .attr("x",130)
229                 .attr("y",290)
230                 .text("")
231                 .attr("fill","blue")
232 
233             let color = "darkorange";
234             svgInDiv.append("path")
235                 .datum(initial)
236                 .attr("id","svgid")
237                 .attr("d", line)
238                 .attr("stroke-width", 3)
239                 .attr("stroke", color)
240                 .attr("fill", color)
241                 .attr("stroke-opacity", 1)
242                 .attr("opacity", 0.5);
```

Isječak koda 9: Prikaz dodavanja path-a i postavljanja inicijalnih vrijednosti

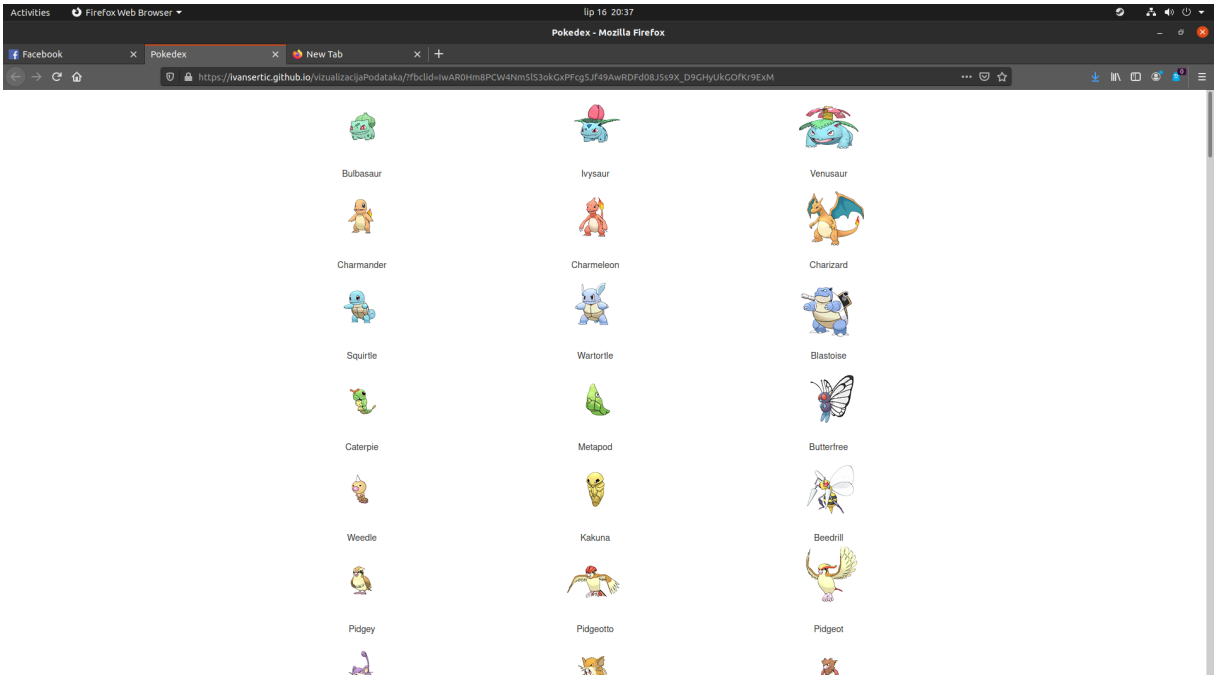
Kako bi se prikazali podatci na klik određenog *Pokemona* unutar funkcije zadužene za obrađivanje klika pozvana je funkcija koja ažurira podatke.

```
241     function plotTheData(data,name) {
242
243         let line = d3.line()
244             .x(d => d.x)
245             .y(d => d.y);
246
247         let color = "darkorange";
248         let coordinates = getPathCoordinates(data);
249
250         console.log(coordinates);
251         var svg = d3.select("body").transition();
252
253         svg.select("#pokeName")
254             .text(name);
255
256         svg.select("#svgid")
257             .attr("d",line(coordinates))
258     }
```

Isječak koda 10: Prikaz funkcije za ažuriranje podataka

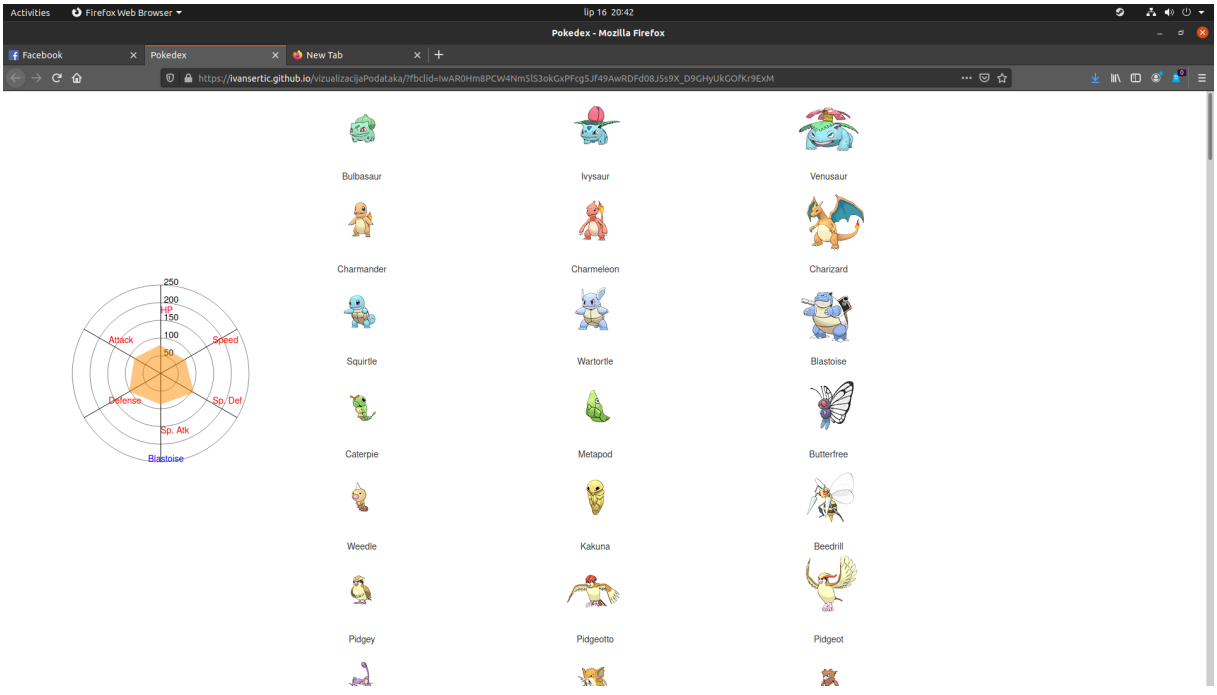
4. Prikaz rješenja

Na stranici su prikazane slike svih *Pokemona* . Kako se prva generacija sastoji od 151 *Pokemona* potrebno je *scrollati* kako bi došli do većine.



Slika 2: Prikaz izgleda stranice

Kako korisnik nebi morao podizati i spuštati stranicu konstantno, *Radar Chart* pojavljuje se s lijeve strane reda u kojemu se odabrani *Pokemon* nalazi.



Slika 3: Prikaz Radar Charta na klik Pokemona

5. Zaključak

Skup podataka o različitim atributima Pokemona nije bio pregledan i teško se znalazilo u njemu. Također same brojčane vrijednosti iako mogu reći koji je atribut bolji od kojeg čovjeku ne znače ništa odnosno teško su čitljive.

Ova vizualizacije postigla je lako pronalaženje želejnog *Pokemona* te prikaz njegovih podataka iz kojeg je lakše vidljivo koliko je koji *Pokemon* balansiran ili koliko je njegove određene statistike prednjače u odnosu na druge.

Svakoko, ova vizualizacija bi se mogla nadograditi većim skupom podataka koji bi uključivao *Pokemone* svih generacije te jedna od ideja koja bi se mogla iplementirati je usporedba statistika dva *Pokemona*.

Literatura

[1] JavaScript, 2020. [Mrežno] Dostupno:https://en.wikipedia.org/wiki/JavaScript

[2] D3.js, 2020. [Mrežno] Dostupno:https://d3js.org/

[3] Bootstrap, 2020. [Mrežno] Dostpuno:https://getbootstrap.com/

[4] CSVJSON, 2020. [Mrežno] Dostupno: https://csvjson.com/csv2json

[5] Alberto Barradas, „Pokemon with stats” [Mrežno] Dostupno: https://www.kaggle.com/abcsds/pokemon/data

[6] Vishal Subbiah, „Pokemon Image Dataset” [Mrežno] Dostupno:https://www.kaggle.com/vishalsubbiah/pokemon-images-and-types

Dostupnost rješenja

Web aplikacija dostupna je na poveznici :
https://ivansertic.github.io/vizualizacijaPodataka/

Izvorni kod aplikacije dostupan je na poveznici: https://github.com/ivansertic/vizualizacijaPodataka