2022ko azaroaren 13a

WEB SISTEMAK PRAKTIKA:

WHO ARE YOU?

Kimetz Egiazabal
Ander Rubio

AURKIBIDEA

*Oharra: Dokumentu honetan ez dira 1, 2 eta 4 ariketak agertzen. Izan ere, ariketa hauetan ikasleoi ez zaigu galderarik egiten.

3.1. Ariketa

1. JQ erabili dugu JSON fitxategitik 2014 ID-a duen elementua lortzeko. Jarraian aurkituko dituzun ariketetan gauza bera egitea eskatzen zaizu, baina oraingo honetan JavaScript erabiliz. Has gaitezen lehenengo ariketarekin: fetch funtzioaz JSON fitxategia lortu eta jarraian filtratu id=2014 duen objektua. Argibidea: filter() metodoa erabili dezakezu.

```
fetch('http://api.football-data.org/v4/competitions')
.then(r => r.json())
.then(data => console.log(data.competitions.filter(item => item.id==2014)))
```

2. Gure hurrengo helburua, Big5 ligak iragaztea izango da. Lehenengo eta behin, soilik TIER_ONE (maila goreneko ligak) aterako ditugu.

```
$ jq -e '.competitions[] | select(.plan == "TIER_ONE")' competitions.json
...
(13 emaitza lortu behar dira).
...

$ jq -e '[.competitions[] | select(.plan == "TIER_ONE")] | length '
    competitions.json
13
```

```
fetch('http://api.football-data.org/v4/competitions')
.then(r => r.json())
.then(data => console.log(data.competitions.filter(item => item.plan=="TIER_ONE")))
```

3. Zein izango litzateke Espainako ligak lortzeko kodea? (JS erabiliz)

```
fetch('http://api.football-data.org/v4/competitions')
.then(r => r.json())
.then(data => console.log(data.competitions.filter(item => item.area.code=='ESP')))
```

4. Zein izango litzateke ESP, DEU, ENG, ITA, FRA herrialdetako TIER_ONE ligak lortzeko kodea? (JS erabiliz)

```
fetch('http://api.football-data.org/v4/competitions')
.then(r => r.json())
.then(data => console.log(data.competitions.filter(item =>
    (item.area.code=='ESP' || item.area.code=='DEU' || item.area.code=='ENG' ||
     item.area.code=='ITA' || item.area.code=='DEU' || item.area.code=='FRA') &&
    item.plan=="TIER_ONE")))
```

5. Aurreko ariketan arazo bat gertatuko da: emaitzan nahi ez dugun (Big5-ekoa ez den) liga bat sartu da (Championship izenekoa).

```
0: {id: 2016, area: {}, name: 'Championship', code: 'ELC', type: 'LEAGUE'
   , ...}
1: {id: 2021, area: {}, name: 'Premier League', code: 'PL', type: 'LEAGUE
   ', ...}
2: {id: 2015, area: {}, name: 'Ligue 1', code: 'FL1', type: 'LEAGUE',
   ...}
3: {id: 2002, area: {}, name: 'Bundesliga', code: 'BL1', type: 'LEAGUE',
   ...}
4: {id: 2019, area: {}, name: 'Serie A', code: 'SA', type: 'LEAGUE', ...}
5: {id: 2014, area: {}, name: 'Primera Division', code: 'PD', type: '
   LEAGUE', ...}
length: 6
```

```
fetch('http://api.football-data.org/v4/competitions')
.then(r => r.json())
.then(data => console.log(data.competitions.filter(item =>
    (item.area.code=='ESP' || item.area.code=='DEU' || item.area.code=='ENG' ||
     item.area.code=='ITA' || item.area.code=='DEU' || item.area.code=='FRA') &&
    item.plan=="TIER_ONE" && item.name!='Championship')))
```

6. map() komandoa erabili aurreko fetch emaitzatik liguen ID-ak lortzeko. Emaitza hau izan beharko litzateke: [2021, 2015, 2002, 2019, 2014]

```
fetch('http://api.football-data.org/v4/competitions')
.then(r => r.json())
.then(data => console.log(data.competitions.filter(item =>
    (item.area.code=='ESP' || item.area.code=='DEU' || item.area.code=='ENG' ||
     item.area.code=='ITA' || item.area.code=='DEU' || item.area.code=='FRA') &&
    item.plan=="TIER_ONE" && item.name!='Championship').map(a => a.id)))
```

5. Ariketa

Ariketa honen enuntziatuan ematen zaizkigun komandoekin erroreak lortzen ditugu. Hauek konpontzeko, *“Arsenal FC”* beharrean *”Arsenal FC*\” idatzi dugu. Honela:

```
cat premier.json| jq -e '.teams[] | select( .name == \"Arsenal FC\")'
```

```
cat premier.json | jq -r '.teams[] | select( .name == \"Arsenal FC\") | .squad[].name'
```

```
cat premier.json | jq -r '.teams[] | select( .name == \"Arsenal FC\") | .squad[1]'
```

5.2.Ariketa

Ez dugu 5.1 ariketaren aipamenik egin komandoa ematen zaigun bezala kopiatu besterik ez delako egin behar. 5.2. ariketan, berriro ere, aurretik aipatu dugun bezala \ txertatu behar izan dugu “-en aurretik erroreak ekiditeko. Honela:

```
cat premier.json| jq -r '.teams[] | .squad[] += { "teamId" : .id, "leagueId":
2021} | .squad |flatten[] | with_entries(if .key == \"dateOfBirth\" then .key = \"birthDate\" else .
end)'
```

5.3.Ariketa

Aurrekoetan bezala, enuntziatuko komandoa hartu dugu, eta “-en aurretik \ txertatu dugu. Honela:

```
cat premier.json| jq -r '.teams[] | .squad[] += { "teamId" : .id, "leagueId":
2021} | .squad |flatten[] | with_entries(if .key == \"dateOfBirth\" then .key=\"birthDate\" else .
end) | with_entries(if .key == \"position\" and .value==\"Offence\" then .value=\"FW\" else if
.key == \"position\" and .value==\"Midfield\" then .value=\"MF\" else . end end)'
```

Hori Office eta Midfield-en aldaketarako litzateke. Patroi berdina jarraituz GoalKeeper eta Defence aldatzeko hau litzateke komandoa:

```
cat premier.json| jq -r '.teams[] | .squad[] += { "teamId" : .id, "leagueId":
2021} | .squad |flatten[] | with_entries(if .key == \"dateOfBirth\" then .key=\"birthDate\" else .
end) | with_entries(if .key == \"position\" and .value==\"Goalkeeper\" then .value=\"GK\" else
if .key == \"position\" and .value==\"Defence\" then .value=\"DF\" else . end end)'
```

5.4.Ariketa

Ariketa honetan aurreko ariketak JavaScript erabiliz egitea eskatzen zaigu. Honetarako funtzio bat sortu dugu, honakoa:

```
function bihurtu(premier){
   premier.teams.forEach(taldea => {
       taldea.squad.forEach(jokalaria => {
           jokalaria.birthdate = jokalaria.dateOfBirth
           delete jokalaria.dateOfBirth
           jokalaria.teamId = taldea.id
           jokalaria.leagueId = premier.competition.id
           let posizioa = jokalaria.position
           if(posizioa=="Goalkeeper"){
               posizioa="GK"
           }
           else if(posizioa=="Defence"){
               posizioa="DF"
           }
           else if(posizioa=="Midfield"){
               posizioa="MF"
           }
           else if(posizioa=="Offence"){
               posizioa="FW"
           }
           jokalaria.position = posizioa
       })
   });
   return premier
}
```

ARIKETA GEHIGARRIAK

11. Ariketa

Ez dugu ariketa gehigarri hau egin.

12. Ariketa

Ariketa gehigarri honen bitartez bezeroak egindako saiakeren egoera *localStorage*-ean gordeta gelditukо da eta ondorioz, orria birkargatu edo itxi eta denbora batera bueltatu arren egindako saiakerak gordeta geratuko dira. Orriak itxi aurretik zuen egoera bera izango du.

Honetarako hainbat aldaketa burutu ditugu kodean:

main.js fitxategiaren aldetik, *game* atributua sortu beharrean *getGame(what)* metodoaren bitartez hasieratu dugu *localStorage*-ean. Ondorioz, game aldagai hau eguneratzeko edo orrialdea kargatzeko *updateGame(what, game)* eta *initializeGame(game)* metodoak erabili ditugu.

stats.js fitxategian, berriz, *getGame* eta *updateGame* metodoak definitu ditugu. Lehena, *game* aldagaia *localStorage*-ean hasieratzeko eta bigarrena *game* aldagaiaren aldaketak eguneratu ahal izateko. *updateGame* motodoan soluzioaren aldaketak duen eragina ere implementatu dugu. Hau da, jokalari berria asmatu behar denean aurreko saiakerak borratu egingo dira berriro jokatu ahal izateko.

Azkenik *rows.js* fitxategian *initializeGame* funtzio bat definitu dugu orrialdea kargatzen denean *localStorage*-ean gordetako saiakerak orrialdean kargatu ahal izateko.

13. Ariketa

13.1.Ariketa

Ariketa gehigarri hau egin dugu. Erlaitz bakoitzaren ondoan favicon-a agertzeko, honakoa egin dugu.

Hasteko, *png* irudi bat jaitsi dugu (*icono.png* gure kasuan), eta hau proiektura igo dugu. Hau erlaitzetan agertzeko, *index.html*-aren *head*-ean honakoa gehitu dugu:

```
<link rel="icon" href="icono.png">
```

13.2.Ariketa

Ariketa gehigarri hau egin dugu. Honetan jokalarien elastikoaren zenbakia 6.atribututzat kontuan hartzea eskatzen zaigu. Honetarako, *rows.js*-n honako aldaketa hauek egin ditugu:

Hasteko, *atribs* array-ari bukaeran *'number'* atributua gehitu diogu. Ondoren, *check* funtzioan, *birthdate*-ekin egiten genuen bezala, aparteko baldintza baten bidez asmatu beharreko jokalariaren zenbakia erabiltzaileak sartu duen jokalariarena baino altuagoa edo baxuagoa den aztertu dugu, erantzunean ↑ edo ↓ jarri jakiteko.

Zenbakia altuagoa edota baxuagoa den azterketa honekin jarraituz, *setContent* funtzioan ere *birthdate*-ekin egiten genuen berbera egin dugu *number*-ekin.