Dokumentacija - Šah, Računarski prakitkum 2

Dora Raštegorac, Vinko Lovrenčić i Luka Jandrijević

July 2021

1 rules.js, play.html

1.1 Varijable

Na početku koristimo angular kako bi stvorili zasebne "box"-ove odnosno šahovska polja. Pomoću toga sada imamo box-x-y gdje x i y označavaju koordinate tog polja.

Na početku `$(document).ready(function())`, na liniji 12, definiramo `chessPieces` u kojima se nalaze slike određenih figura. Slike figura prikazujemo HTML kodom. Tako na primjer string "&#9812" označava sliku bijelog kralja i tako dalje...

Varijabla `player` je zadužena za pratiti koji igrač je trenutno na redu, bijeli ili crni. Na početku je jednaka white jer je na početku bijeli na potezu. Varijabla `promotion` će nam pomoći da postavimo potrebne podatke pri promociji pijuna.

Pomoću varijable `select` pratimo može li se zadana figura micati, tip figure te pozicija figure, odnosno u kojem box-u se nalazi.

1.2 Postavljanje ploče

Na liniji 48, pozivamo funkciju koja je zadužena da polju s koordinatama i i j pridruži klasu `dark-box`, odnosno `light-box` ovisno o tome je li `i+j` paran ili neparan broj. Time dobivamo šahovnicu. S funkcijom `setNewBoard(box,i,j)` na polje sa koordinatama (i, j) postavljamo figuru ako se na početku igre tamo treba prikazati ta figura. Figuru postavljamo na zadano polje s funkcijom `setPiece(box,color,type)`. Na kraju još pozivamo funkciju `setTheme()` koja pomoču jQuery-a postavlja `light-box` i `dark-box` na zadanu boju.

1.3 Click event funkcije

Nakon postavljanje ploče definiramo funkciju koja je zadužena za click evente. Prvo obrađujemo slučaj ako je pijun promaknut. Odabirom od ponuđenih figura, ona se postavlja na potrebnu poziciju i postavlja klasu `placed`.

Kada kliknemo na polje tada se poziva funkcija zadužena za klik na klasu `box`. Ako je taj box, odnosno polje već od prije bilo odabrano to jest, ima klasu `selected`, tada se ta klasa za to polje briše i klasa `suggest` koja označava

moguća polja kojom se ta figura može kretati također briše, a varijablu `select` postavljamo na originalne vrijednosti.

Sljedeće što moramo provjeriti je slučaj ako se figura može micati. U tom slučaju znamo da se u `select.piece` nalaze sve potrebne informacije o figuri, boja i tip. Prvo provjerimo ako je odabrana druga figura iste boje. Ako je, tada odabiremo tu figuru. Figuru možemo pomaknuti samo na ona polja koja imaju klasu `suggest`. Ako taj box ima klasu `suggest`, tada pomoću funkcije `setPiece(box, color, type)` postavljamo tu figuru na zadano polje (box) i brišemo prethodno polje. Varijablu `select` opet vraćamo na originalne vrijednosti te mijenjamo igrača.

1.4 setPiece(box,color,type)

Funkcija `setPiece(box,color,type)` na početku provjerava je li figura "pojela" kralja na tom polju. Ako je, tada se poziva `showWinner()`, odnosno igra je završena. Dodatno, ona još provjerava je li pijun u poziciji za promaknuće. Varijabla `j` će nam reći na kojem rangu se nalazi figura. Ako je pijun u poziciji za promaknuće, tada se igra "zatamnjuje" pomoću `opacity` te pomoću jQuery-a prikazujemo popis figura u koje se pijun može promaknuti. Na kraju jednostavno postavimo HTML tekst u određeni HTML znak figure.

1.5 deleteBox(box)

Ova funkcija briše figuru sa određenog polja. Također briše klasu `placed` i `selected` sa određenog polja.

1.6 getNextMoves(selectedPiece, selectedBox)

Funkcijom `getNextMoves` dohvaćamo moguće poteze odabrane figure. Prvo dohvatimo informacije o figuri i spremimo ih u određene varijable. Zatim na sličan način dohvatimo i koordinate na kojima se ta figura nalazi. Ovisno o tome o kojoj figuri je riječ, pozivamo funkciju koja u varijablu `nextMoves` vraća sve moguće poteze te figure. Toj funkciji prosljeđujemo parametar `offset` koja predstavlja u kojim sve smjerovima se određena figura može micati.

Sastoji se od niza $[x, y]$ pri čemu x predstavlja pomak u okomitom smjeru dok y predstavlja pomak u vodoravnom smjeru. Tako na primjer top (eng. "rook") ima postavljen `offset` na $[[0, 1], [0, -1], [1, 0], [-1, 0]]$ jer se top može po pravilima micati gore,dolje,lijevo i desno. Za pijuna imamo u `offset`-u i $[0, 2]$ odnosno $[0, -2]$ jer kad je pijun na početnom položaju onda se može micati za 2 polja gore odnosno dolje, ovisno o tome o kojoj boji je riječ.

Za konja (eng. knight) `offset` ima niz $[2, -1]$ jer konj se može kretati za 2 polja gore i jedno polje ulijevo.Slično kao i za topa smo postavili i `offset` drugim varijablama.

1.7 pawnMoves(i, j, color, moves)

S ovom funkcijom dohvaćamo moguće poteze pijuna. U `tI` spremamo okomiti smjer a u varijablu `tJ` vodoravni smjer micanja figure. Prolazimo kroz varijablu `offset` te provjeravamo hoćemo li otići izvan granica ploče za odabranu figuru. Ako nećemo, tada provjeravamo još nalazi se na putu neka druga figura (to radimo pomoću klase placed). Sve te moguće poteze spremamo u `nextMoves` te ju nakraju funkcije vraćamo.

1.8 knightMoves(i, j, color, moves)

Ovom funkcijom dohvaćamo moguće poteze konja i kralja. Slično kao i prije, ako se na tom mjestu nalazi protivnikova figura tada ta figura može otići na to mjesto. To provjeravamo sa `indexOf(color)` koja vraća negativnu vrijednost ako ne pronađe index na kojem se nalazi podstring "color".

1.9 otherPiecesMoves(i, j, color, moves)

Ovom funkcijom dohvaćamo moguće poteze ostalih figura (kraljica, lovac, top). Ovdje koristimo varijablu `sugg` pomaže u detektiranju ako se neka figura nalazi na putu mogućih poteza. U tom slučaju gledamo je li ta figura protivnikova ili nije. Ako je protivnikova tada se i tamo možemo micati s tom figurom. U suprotnom, ne možemo jesti svoje figure pa se dalje ne možemo kretati.

1.10 suggestNextMoves

Funkcija pomoću koje za sve legalne poteze dodajemo tim poljima klasu `suggest`.