Gymnázium, Praha 6, Arabská 14

Programování

myChess

Ročníková práce

Duben 2020

Autor: Filip Dubják

Třída: 4.E

Rok: 2019/2020

Vyučující: Lána Jan

Prohlašujeme, že jsem autor tohoto projektu, všechny citace jsou řádně označené a všechna použitá literatura a další zdroje jsou v práci uvedené. Tímto dle zákona 121/2000 Sb. (tzv. Autorský zákon) ve znění pozdějších předpisů udělujeme bezúplatně škole Gymnázium, Praha 6, Arabská14 oprávnění k výkonu práva na rozmnožování díla (§ 13) a práva na sdělování díla veřejnosti (§ 18) na dobu časově neomezenou a bez omezení územního rozsahu.

V Praze dne 10. května 2019 .....

Anotace

Tato ročníková práce se zabývá mým postupem při programování mé ročníkové práce o řešení šachových problémů do tří tahů. Zabývá se problémem v řešení daných šachových pozic. Zabývá se strukturou mého programu a jednoduchým popsáním všech komponentů a funkcí. Zabývá se řešením kombinace JavaScriptových knihoven a problémy, které při psaní vznikly.

Abstract

This year's work deals with my progress in programming my year's work on solving chess problems in three moves. It deals with the problem in solving given chess positions. It deals with the structure of my program and a simple description of all components and functions. It deals with solving a combination of JavaScript libraries and problems that arose during writing.

Obsah

1. Zadání

Cílem ročníkové práce je vytvořit v internetovém prostředí tréninkovou formu šachových cvičení. Aplikace bude umět vyhodnotit nejlepší tahy všech zadaných pozic, které jsou dostupné na internetu. Bude umět zobrazovat šachové pozice a hrát hru šachu podle pravidel, také umožní uživateli hrát šachové pozice, které budou ve stylu mat do tří tahů „proti počítači“. Také bude udržovat záznam o tazích, které při hře proběhnou. V případě toho, že uživatel netáhne správně, aplikace mu oznámí chybu. Také bude možnost toho si nechat poradit. Stránka by měla mít jak mobilní, tak i standardní verzi v internetových prohlížečích.

2. Úvod

Dnes již každý šachysta používá k tréninku různé webové stránky, které jim pomáhají se zlepšovat v daných herních situací, prakticky od *openingu* až po *endgame.* Já sám jsem se rozhodl naprogramovat podobnou verzi tichto stránek, za účelem se zlepšit ve psaní webových stránek, skriptů a také v porozumění serverové strany.

Mým úkolem bylo vytvořit webovou stránku, která by byla tohoto schopna. Rozhodl jsem se, že si zvolím cvičení do tří tahů, které by bylo nejjednoduší na napsání algoritmu, který by vyhledával nejlepší možné tahy.

3. Použité technologie

3. 1. Visual StudioCode

Visual Studio Code je editor zdrojového kódu vyvíjený společností Microsoft pro operační systémy Windows, Linux a macOS. Obsahuje podporu pro Git, zvýraznění syntaxe, kontextový našeptávač a podporu pro ladění a refaktorizaci. Zdrojový kód je svobodný software pod licencí MIT. (Tchoř)

3. 2. GitHub

Pro spravování verzí projektů jsem společně s VCS ve vývojovém prostředí používali GitHub, webovou službu, která poskytuje bezplatný webhosting pro open-source projekty. Svůj projekt jsem musel rozdělit do dvou složek a to *client* a *server*, jelikož jsem musel mýt v *backendové(serverové)* části něco jiného, než na straně klienta.

3. 3. JavaScript

JavaScript je programovací jazyk, který odpovídá specifikaci ECMAScript. Má kudrnaté závorky syntaxe, dynamické psaní, prototypové objektové orientace a prvotřídní funkce.

Spolu s HTML a CSS je JavaScript jednou z hlavních technologií World Wide Web. JavaScript umožňuje interaktivnost na webových stránkách a je nezbytnou součástí webových aplikací.

Jako jazyk s více paradigmatem podporuje JavaScript styly programování založené na událostech, funkční a imperativní. Má aplikační programovací rozhraní (API) pro práci s textem, daty, regulárními výrazy, standardními datovými strukturami a DOM Object Object Model (DOM). Samotný jazyk však neobsahuje žádný vstup / výstup (I / O), jako jsou síťová, úložná nebo grafická zařízení, protože hostitelské prostředí (obvykle webový prohlížeč) poskytuje tato API.

JavaScript enginy byly původně používány pouze ve webových prohlížečích, ale nyní jsou zabudovány do některých serverů, obvykle přes Node.js. Jsou také zabudovány do různých aplikací vytvořených pomocí rámců, jako jsou Electron a Cordova. (Bishonen)

V mém projektu je JavaScript nezbytný. Bez něj bych mohl nad myšlenkou vytvořit šachy online úplně zapomenout.

3. 4. Node.js

Node.js je open-source, multiplatformové běhové prostředí JavaScriptu, které provádí JavaScript kód mimo webový prohlížeč. Program Node.js umožňuje vývojářům používat JavaScript k psaní nástrojů příkazového řádku a pro skriptování na straně serveru - spouštění skriptů na straně serveru k vytváření dynamického obsahu webové stránky před odesláním stránky do webového prohlížeče uživatele. V důsledku toho Node.js představuje paradigma „JavaScript všude“, který sjednocuje vývoj webových aplikací kolem jediného programovacího jazyka, než různé jazyky pro skripty na straně serveru a klienta.

Ačkoli .js je standardní přípona názvu souboru pro kód JavaScript, název „Node.js“ v tomto kontextu neodkazuje na konkrétní soubor a je pouze názvem produktu. Node.js má architekturu řízenou událostmi, schopnou asynchronních *Input*/Output. Cílem těchto konstrukčních možností je optimalizovat propustnost a škálovatelnost ve webových aplikacích s mnoha operacemi vstupu / výstupu a také pro webové aplikace v reálném čase (např. Komunikační programy v reálném čase a hry prohlížečů). (Rfl)

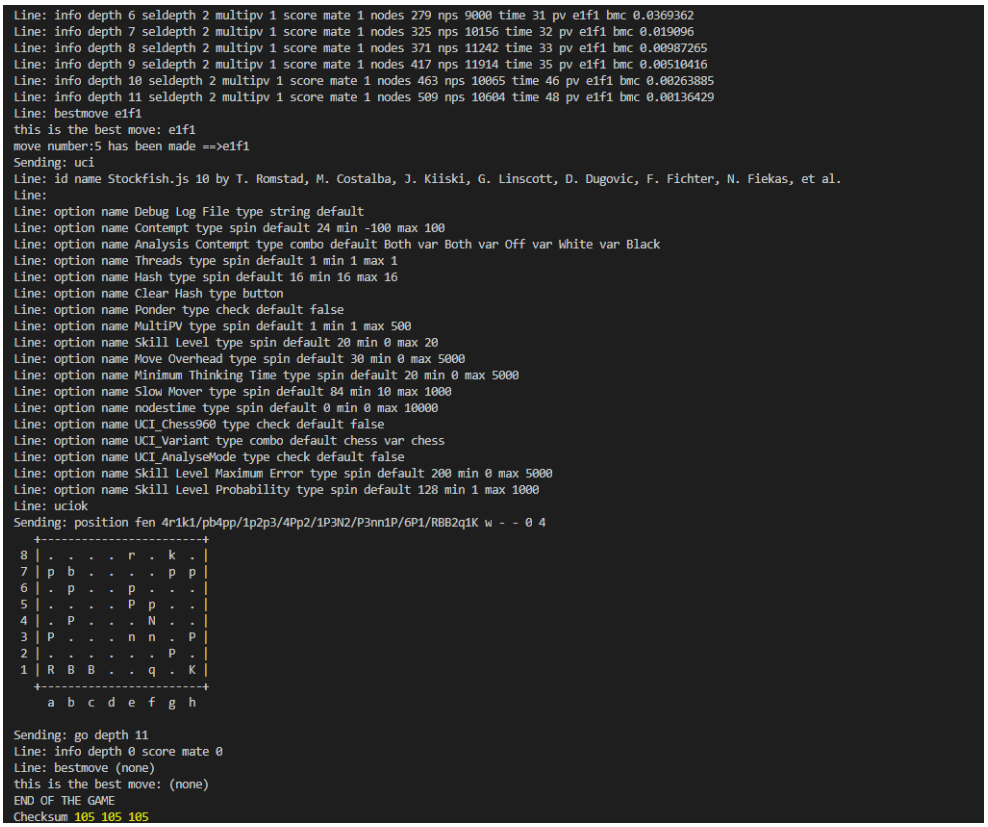
V mém projektu je Node.js naprosto nezbytný. Jednak je potřeba k práci s textem, jako jsou třeba šachové pozice a také je nutný k vyřešení daných pozic a zápisu předpokládaných tahů hráče do souboru na straně klienta. Verzi Node.js jsem si vybral LTS za cíle dlouhodobé kompatibilitě.

Obrázek 1 Node.js / https://upload.wikimedia.org/wikipedia/commons/thumb/d/d9/Node.js_logo.svg/1200px-Node.js_logo.svg.png

4. Instalace

Instalace je velmi jednoduchá, tudíž jsem se jí rozhodl popsat v pár bodech.

1. K instalaci je nejprve potřeba stáhnout a nainstelovat Node.js, který najdete zde **https://nodejs.org/dist/v12.16.2/node-v12.16.2-x64.msi** .
2. Poté si buď přímo stáhněte z této webové stránky source code **https://github.com/UglyBoySadFace/Chess_de_la_mate** , nebo pokud máte git, tak si pomocí příkazu ***git clone https://github.com/UglyBoySadFace/Chess_de_la_mate.git*** stáhněte kód do vámi zvoleného adresáře.
3. Zadáme příkaz ***cd Chess_de_la_mate*** a ten nás přenese do námi vytvořené složky, ve které se nachází složka server a client.
4. Dále se pomocí příkazu ***cd server,*** dostaneme do složky se serverem.
5. Pro spuštění aplikace vždy používáme příkaz ***npm start***, který za pomoci nainstalovaných modulů nastartuje naši aplikaci.



Obrázek 2 CLI ze spuštěného programu

5. Knihovny

K ulehčení mnoha věcem a k celkové přehlednosti kódu jsem použil pár knihoven. Za pomocí knihovny **Chess.js** řeším vše co se týče samotné hry šachu, a to pozice figurek při začátku, dostupná místa kam se může figurkou táhnou, načítání pozic ve formátu FEN, ale i png, to všechno je jen datová stránka mého projektu, pro zobrazení těchto dat používám další knihovnu a to **Chessboard.js**. Ta za pomocí dat, které získá z **Chess.js** vygeneruje šachové pole s figurkama ve hře. Také umožňuje hráči pohybovat figurkami, díky jednoduché funkce, kterou rozepíši později v doumentaci. Dále jsem potřeboval knihovnu **jQuery**, která dělá psaní v JavaScriptu mnohem jednodušší, věci jako procházení a manipulace s HTML dokumenty, manipulace s událostmi, animace apod.

6. Backend

6. 1. Express

Nejprve bylo zapotřebí vyrenderovat stránku clientovi, na to jsem použil Express, který umožňuje širokou škálu funkcí.

- Umožňuje nastavit middlewares tak, aby odpovídaly na HTTP požadavky.

- Definuje směrovací tabulku, která se používá k provádění různých akcí na základě metody HTTP a adresy URL.

- Umožňuje dynamicky vykreslit stránky HTML na základě předávání argumentů do šablon.

Kód pro nastavení cest, je velmi krátký, jelikož se můj web skládá pouze z jedné html stránky. Pomocí express ještě řeším chyby, které by nastaly, kdyby se chtěl někdo dostat na stránku, která neexistuje.

```
//set up routes
router.get('/', function (req, res) {
  res.sendFile(path.join(client + '/index.html'));
});


app.post('/', function(req,res){

})
app.use('/', router);
app.use(express.static(client));
app.use(function(req, res, next) {
  var err = new Error('Not Found');
  err.status = 404;
  next(err);
});

if (app.get('env') === 'development') {
  app.use(function(err, req, res, next) {
      res.render('error', {
          message: err.message,
          error: err
      });
  });
}
```

Obrázek 3 RoutesSetUp

6. 2. FileSystem

Pomocí FileSystému v projektu řešíme práci se soubory, jelikož na internetu aktuálně neexistuje žádné API, které by mohlo pomoc ve vyhledávání pozic, musel jsem použít textový soubor se zhruba pětisty pozicemi, které jsem našel na internetu. Tedy, pomocí FileSystému pracuji se souborem **m8n3.txt,** který obsahuje zápis pozic a hráče, jež jí dosáhly. Pozice se opakovala každý pátý řádek, tudíž jsem každý pátý řádek vzal a uložil jsem ho do pole s pozicemi, pokud tato pozice byla hratelná. Pomocí FileSystému později posílám klientovi vyřešené pozice.

```
//vytvori fenpos.json s moznymi pozicemi, ktere jsou validni
//pri pridani jine database je treba toto upravit
fs.readFile("m8n3.txt", function (err, buf) {
  var puzzle = buf.toString().split(/\r?\n/);
  var arrayOfPositions = '';
  var Fen;
  // i = starting line, i = i + k{spaces between lines}
  for (var i = 9; i < puzzle.length; i = i + 5) {
    arrayOfPositions += puzzle[i] + "\n";
    Fen = arrayOfPositions.split('\n');
  }
  delete arrayOfPositions;
  arrayOfPositions = [];
  for (i = 0; i < Fen.length; i++) {
    if (chess.load(Fen[i])) {
      arrayOfPositions.push(Fen[i])
    }
  }
```

Obrázek 4 PositionCheck

6. 3. Vyhodnocování tahů

Zprva jsem plánoval, že bych si napsal algoritmus vlastní, bohužel jsem to ale nezvládl. Na to abych vytvořil vlastní algoritmus vyhodnocování bych potřeboval hodně času a nápad toho, že bych zkoušel všechny možné tahy, mi přišel stupidní.

Tudíž jsem zapátral po internetu a našel jsem velmi dobrý *opensource engine*, který používá valná většina šachových stránek. Engine se jmenuje **Stockfish**. **Stockfish** je trvale zařazen na první nebo v horní části většiny seznamů šachových *enginů* a je nejsilnějším konvenčním šachovým *enginem* na světě.

Jediný problém se **Stockfishem**, byl to, že nemá použitelnou verzi napsanou v JavaScriptu, **Stockfish** je totiž napsaný v jazyku C++, musel jsem tedy najít verzi, která je emulovaná, což se mi nakonec povedlo. **Stockfish** pracuje jako CLI(Command Line Interface). Tudíž jsem nastavil odesílání a přijímání zpáv z jeho CLI a začal jsem ho nechat hrát proti sobě. Výsledné nejlepší tahy jsem pak zapisoval do dalšího pole s názvem solutions.

Při tomto nastal další problém a to, že některé z pozic se nedaly nahrát do **Stockfishe** a některé se nedaly nahrát do knihovny **Chess.js**, mohl za to právě špatně napsaný dokument s pozicemi, který jsem našel na internetu. Pozice **FEN(Forsyth–Edwards Notation)**, byla totiž odbyle zapsaná, a tudíž fungovaly akorát pozice, ve kterých začínal hrát černý. Dále jsem potřeboval kontrolovat, jestli pozice skutečně končí do tří tahů.

Ve výsledku bylo hratelných 115/500 pozic.

```
function check_forBestMove(position_of_puzzle) {
  send('uci');
  engine.onmessage = function (line) {
    var match;
    console.log("Line: " + line)

    if (typeof line !== "string") {
      console.log("Got line:");
      console.log(typeof line);
      console.log(line);
      return;
    }

    if (line === "uciok") {
      send("position fen " + position_of_puzzle);
      console.log(chess.ascii());
      //for mate in 5 we will need to search somewhere betweeen 10-11 plies
      send('go depth 11');
    }
    if (line.indexOf("bestmove") > -1) {
      match = line.match(/bestmove\s+(\S+)/);
      if (match) {
        bestmove = (match[1]);
        console.log('this is the best move: ' + bestmove);
        if (!chess.in_checkmate()) {
          if (curmoves === 6) {
            getRidOf();
            positionsbefore--;
          } else {
            moveAndCollect();
            check_forBestMove(chess.fen());
          }
        } else {
          console.log('END OF THE GAME');
          counter++;
          getBestMovesOfGame(counter);
        }
      }
    }
  }
}
```

Obrázek 5 checkForBestMove

7. Frontend

7. 1. Struktura projektu

Frontend mé aplikace jsem se snažil udělat co nejjednodušší. Vytvořil jsem tedy soubor **index.html**. Do tohoto souboru jsem se rozhodl napsat všechno co může moje stránka poskytnout uživateli.

Z důvodu toho, že je docela komplikované linkovat soubory JavaScriptu, které navíc obashují knihovny, se souborem .html, rozhodl jsem se mít vše pohromadě. V **index.html** tedy najdete jak hypertextový kód stránky, tak i skripty a kaskádové styly.

7. 1. Vzhled stránky

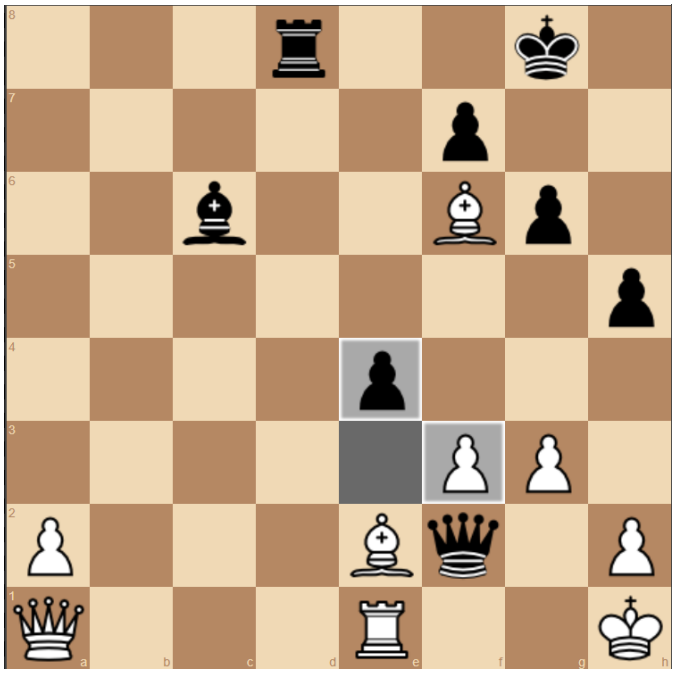
Vzhled stránky je přehledný a jednoduchý, na levé straně se nachází box s údajemi, které se právě nacházejí na šachovnici a informace o tom, kdo je na tahu. Pod boxou a v boxe se nachází skupina tlačítek, které interagují se šachovnicí. Tlačítko „GO back“ zajišťuje zpětný pohyb hráče. Tlačítko „Clear Board“ se zbaví všech figurek na šachovnici. Pomocí tlačítka „Start random puzzle“ se načte jedna z pozic, ve které má hráč za úkol dát mat do tří tahů. Tlačítko „Hint“ hráči indikuje, jaký je jeho nejlepší tah. Když chce hráč hrát, stačí najet na figurku, držet levé tlačítko myši a přetáhnout ji na vybranou pozici.

Obrázek 6 Vzhled stránky

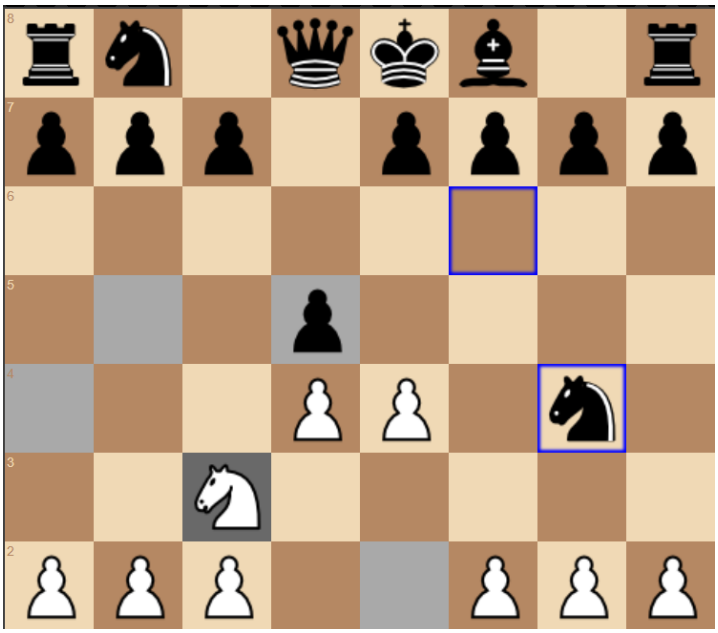
7. 2. Vzhledové vylepšení

Připadalo mi, že hře něco chybí, tak jsem se rozhodl, zvýrazňovat poslední odehraný tah každého hráče. Navíc jsem se rozhodl, že bych pro méně zkušené ve hře šachu, udělal menší návod na to s čím se kam táhne, tudíž jsem zvýraznil všechna dostupná políčka pro figurku po najetí myší.

Obrázek 7 Nápověda při hře



Obrázek 8 Zvýraznění posledně odehraných tahů



Obrázek 9 Zvýraznění dostupných míst

8. Závěr

Tento projekt byl pro mě velmi zajímavým přínosem, zároveň mě mrzí, že jsem na něm nestrávil víc času, a že na internetu nejsou k dispozici volně pozice. Ve vývoji bych i nadále rád pokračoval. Nejdřív bych začal s *real time* vyhodnocováním tahů, tak že bude moc uživatel hrát proti *enginu*, na bázi toho, že každý jeho tah bude napsán do formy a odeslán na server a posléze vyhodnocen. Dále bych vyřešil ten problém s málo dostupnými pozicemi, našel jsem jednu stránku, která ale bohužel neměla doposavaď vyřešený problém s jejich API **https://chessblunders.org/faq** . Také bych v budoucnu přidal možnost se registrovat a sledovat jak se jim zatím v šachových cvičeních daří.

9. Zdroje

Works Cited

Bishonen. JavaScript. 28. 4 2020. <https://en.wikipedia.org/wiki/JavaScript>.

Rfl. Wikipedia. 24. 4 2020. <https://en.wikipedia.org/wiki/Node.js>.

Tchoř. „Visual Code Studio.“ 23. 4 2020. wikipedia. 28. 4 2020 <https://cs.wikipedia.org/wiki/Visual_Studio_Code>.

Obrázek 1: Node.js. Dostupné z URL: <https://upload.wikimedia.org/wikipedia/commons/thumb/d/d9/Node.js_logo.svg/1200px-Node.js_logo.svg.png >