Piškvorky (Gomoku) a Min-Max

## Marek Chwistek – Maturitní práce 2023

Contents

[Marek Chwistek – Maturitní práce 2023 1](#_Toc127705240)

[1. Úvod 3](#_Toc127705241)

[2. Dokumentace 3](#_Toc127705242)

[I. HTML 3](#_Toc127705243)

[II. JavaScript 4](#_Toc127705244)

[Globální proměnné 4](#_Toc127705245)

[Interaktivita okna 5](#_Toc127705246)

[function squareClickEvent 5](#_Toc127705247)

[class Board 6](#_Toc127705248)

[class Square 9](#_Toc127705249)

[function chainLength 10](#_Toc127705250)

[function reverseDir 10](#_Toc127705251)

[function evaluatePosition 11](#_Toc127705252)

[function emptyAround 11](#_Toc127705253)

[function evalChains 12](#_Toc127705254)

[function arraySum 12](#_Toc127705255)

[function reset 12](#_Toc127705256)

# Úvod

Ve své práci jsem se rozhodl vytvořit hratelné piškvorky na 5, také známé jako „Gomoku“.  
Hrát sám proti sobě ovšem není zábava, a tak jsem přidal soupeř€ - MinMax algoritmus, který hraje proti nám.

Celý program je napsaný v JavaScriptu, s velmi jednoduchou nástavbou HTML – to je zde využito jako velice jednoduché uživatelské rozhraní.

# Dokumentace

## HTML

Soubor index.html

Soubor „index.html“ je opravdu pouze jednoduchá kostra – připraví místo pro zobrazování bodů každého hráče a canvas, na který budeme následovně kreslit vše co se bude ve hře dít.

Výška a šířka canvasu jsou zde nepodstatné – musí se nastavovat až ve skriptu, na který je v řádku 10 odkázáno.

A screenshot of a computer

Description automatically generated with medium confidenceChart

Description automatically generated

## JavaScript

JavaScript, duše projektu.

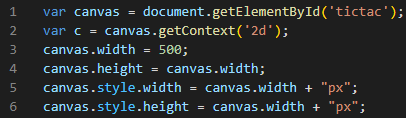
Vzhledem k malému rozsahu projektu je celá logika programu v jednom souboru.

Soubor index.js

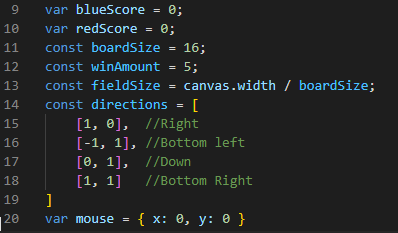
Nejdříve se koukněme na konfiguraci – globální proměnné a co dělají

### Globální proměnné

I přestože bychom je moc neměli používat, bez několika se neobejdeme.



V první sérii proměnných si připravuji canvas. Požadovanou velikost, zajišťuji jeho tvar čtverce a proměnná „c“ mi následně umožní na něj kreslit.



Následně pár logických proměnných – blueScore a redScore sledují výhry každého z hráčů, boardSize určuje velikost hracího pole, winAmount potřebné množství stejného symbolu v řadě pro výhru.

fieldSize je zde opět kvůli kreslení – spočítá jak velké má být každé políčko

directions je pole „tupleů“, které určují různé směry od políčka, ve formáty [x, y], přičemž levý horní roh je [0, 0], dolů se zvedá y a doprava se zvedá x.

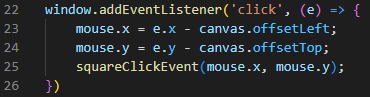
nakonec proměnná mouse ukládá pozici myši.



Poslední globální proměnnou v kódu je board – objekt herní desky, ke kterému se dostaneme později.

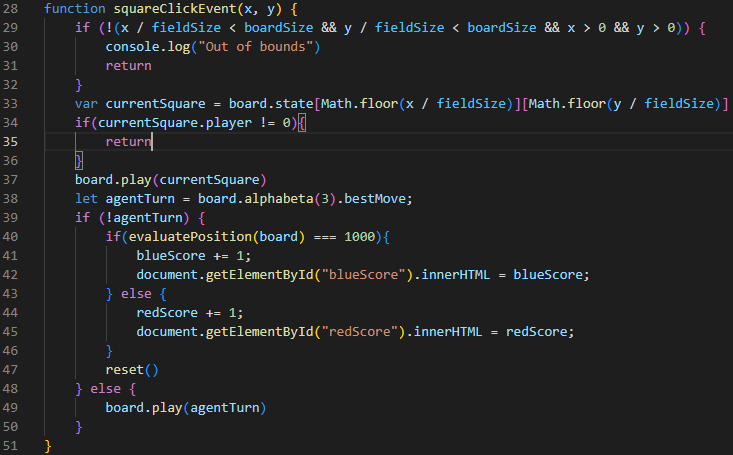
### Interaktivita okna

Hra se dá ovládat pouze klikáním myši, zajištěné eventListenerem, který nám poskytuje html.



Při každém kliknutí do okna zavolá fci squareClickEvent s x, y souřadnicemi kurzoru.  
Řádky 23 a 24 pouze ošetřují situaci, kdy by okno bylo posunuto.

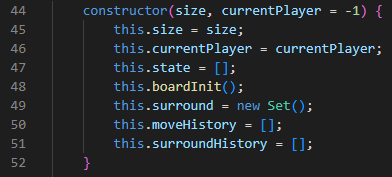
### function squareClickEvent



Tato funkce je zavolána při každém kliku myši.

Nejdříve na řádcích 29-32 zkontroluje, zda-li bylo kliknuto na desku.  
Řádek 33 spočítá, na jaké políčko uživatel vlastně klikl.  
Dále zahraje na toto políčko, a řekne algoritmu aby odpověděl na tento tah.  
Pokud agent nenajde tah, protože někdo už vyhrál, restartuje desku. A přičte skóre

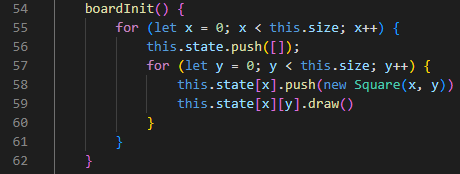
### class Board

Třída desky – pamatuje si celý stav desky, její minulost a hráče na tahu.

constructor, sestrojí instanci třídy s vlastnostma.

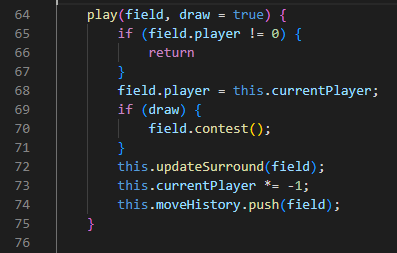
Surround obsahuje políčka vedle již zaplněných políček, surroundHistory jejich minulost

#### Metoda boardInit



Nainicializuje úvodní stav (state[]) desky. 2d pole plné instancí třídy Square a vykreslí je

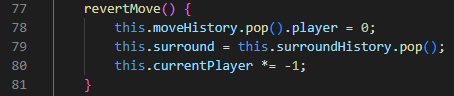
#### Metoda play



Zahraje tah na políčko field, pokud je prázndé, podle toho jaký hráč je na tahu a změní hráče na tahu. Také si zapamatuje tah.

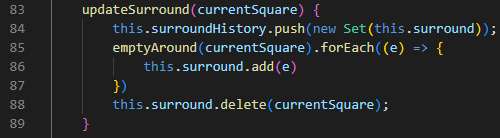
Pokud není řečeno jinak, vykreslí jej na desku.

#### Metoda revertMove



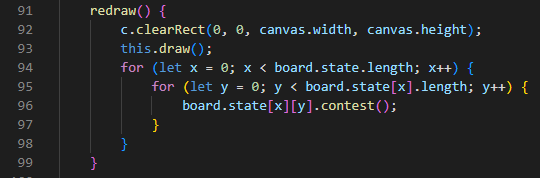
Vrátí zpět poslední tah

#### Metoda updateSurround



Přidá do surround všechna prázdná políčka kolem daného políčka a odstraní toto políčko z okolních, protože bylo právě zaplněno

#### Method redraw

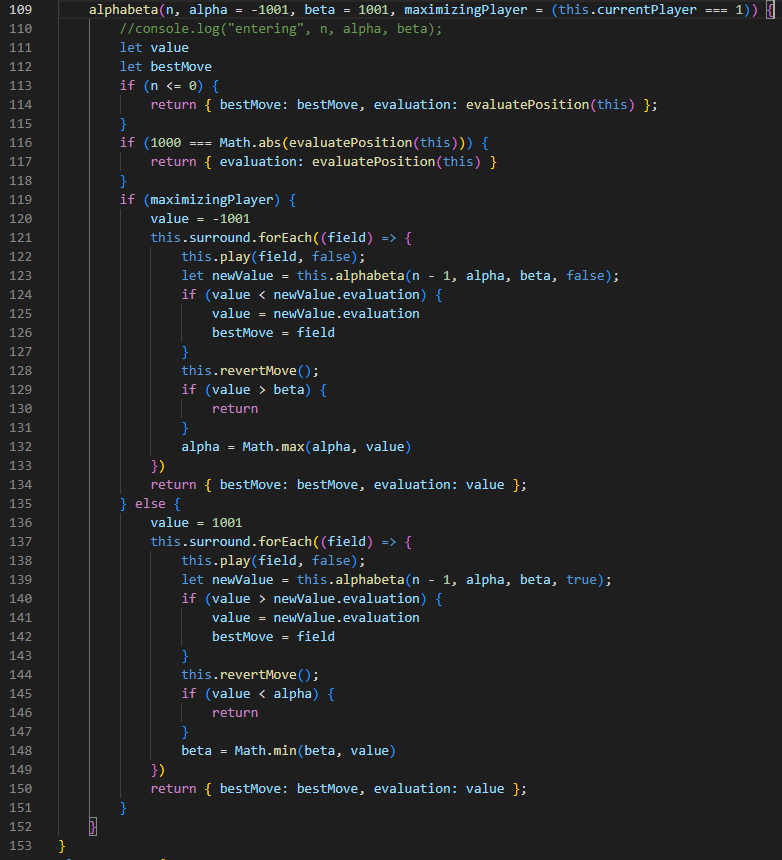


Znovu vykreslí celou desku

#### Method draw

Vykreslí okraje každého políčka

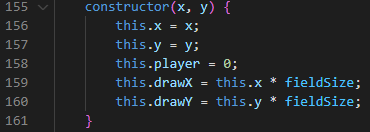
#### Method alphabeta



Aplikuje MinMax s Alpha-beta prořezáváním. Použitá hloubka je 3

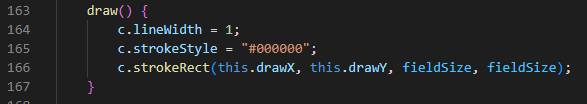
### class Square

Třída políčka.



Vlastnosti x, y – souřadnice v rámci kódu  
drawX drawY, souřadnice pro kreslení  
player – čí je políčko

#### Method draw



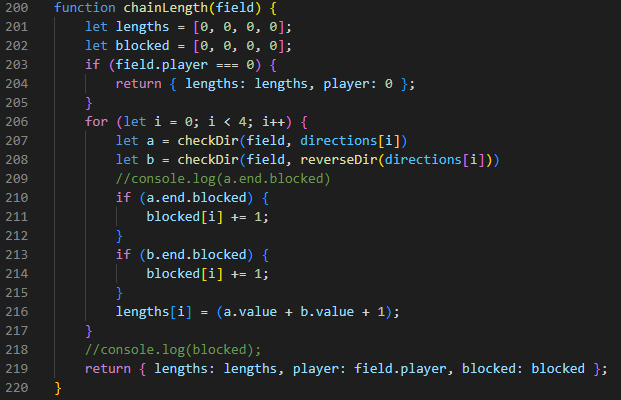
Obkreslí políčko

#### Method contest



Podle toho jaký hráč vlastní políčko, naplní jej buď křížkem nebo kolečkem

### function chainLength



Zkontroluje všechny řetězce, kterých je toto políčko součástí a vrátí, jak dlouhý každý z nich je a jestli jsou na konci zablokovány cizím políčkem či okrajem desky

### function reverseDir



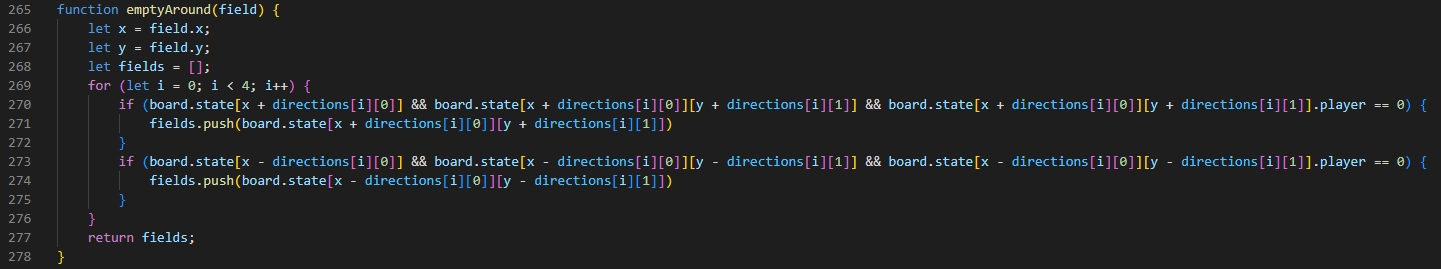
Vrátí opačný směr

### function evaluatePosition



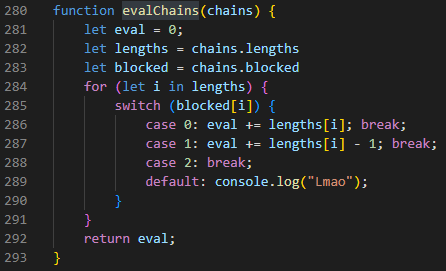
Zhodnotí pozici, vrátí hodnotu mezi -1000 a 1000, kde -1000 křížek vyhrál a 1000 kolečko vyhrálo

### function emptyAround

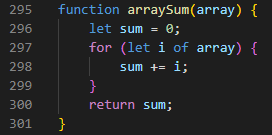


Vezme políčko, koukne se kolem a vrátí pole s prázdnýma políčkama kolem vstupního

### function evalChains

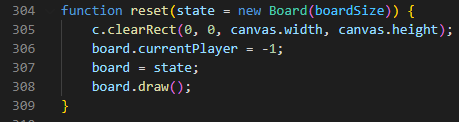
  
Zhodnotí hodnotu daného pole řetězů a informací o tom, jak moc jsou blokovány.   
Blokován z jedné strany je o 1 horší, z obou je bezcenný.

### function arraySum



Sečte pole a vrátí sumu

### function reset



Nastaví desku na začáteční hodnoty