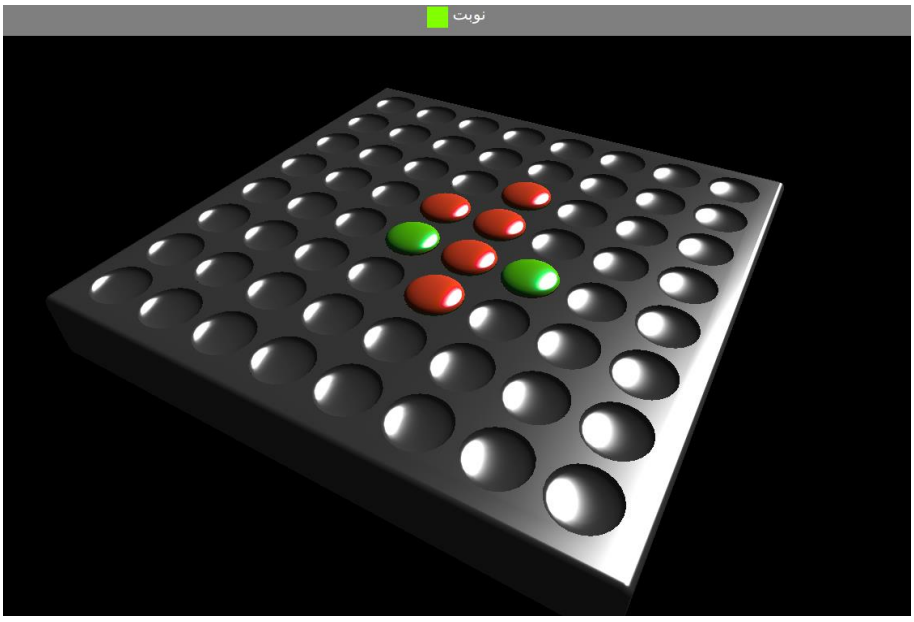
وزارت علوم تحقیقات و فناوری

دانشگاه علم و فرهنگ

مستند پروژه درس گرافیک کامپیوتری

موضوع پروژه:

بازی اوتللو



طراح:

محمد رضا واحد

مدل سازی:

برای پیاده سازی این پروژه، ابتدا اشیای آن توسط نرم افزار Cinema٤D مدل‌سازی شد. سپس از ۳ شیئ مهره، صفحه بازی و جای مهره ها به صورت جداگانه خروجی .obj گرفته شد تا قابلیت استفاده آن در Threejs وجود داشته باشد. همچنین از متریال های آن یک خروجی با پسوند .mtl گرفته شد.

ساخار فایل html:

درون body فایل html دو عنصر قرار دارد. Header که بالای صفحه قرار دارد جهت نمایش نوبت بازیکن و canvas که بازی در آن نمایش داده می‌شود.

index.html

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>Othello</title>
    <link rel="stylesheet" href="css/styles.css">
</head>
<body>
    <header>
        <div id="alternate">
            نوبت
            <div id="alternate-color" class="red"></div>
        </div>
    </header>

    <canvas id="c">

    </canvas>
</body>
<script src="js/three.min.js"></script>
<script src="js/LoaderSupport.js"></script>
<script src="js/OBJLoader2.js"></script>
<script src="js/MTLLoader.js"></script>
<script src="js/main.js"></script>
</html>
```

کد های برنامه:

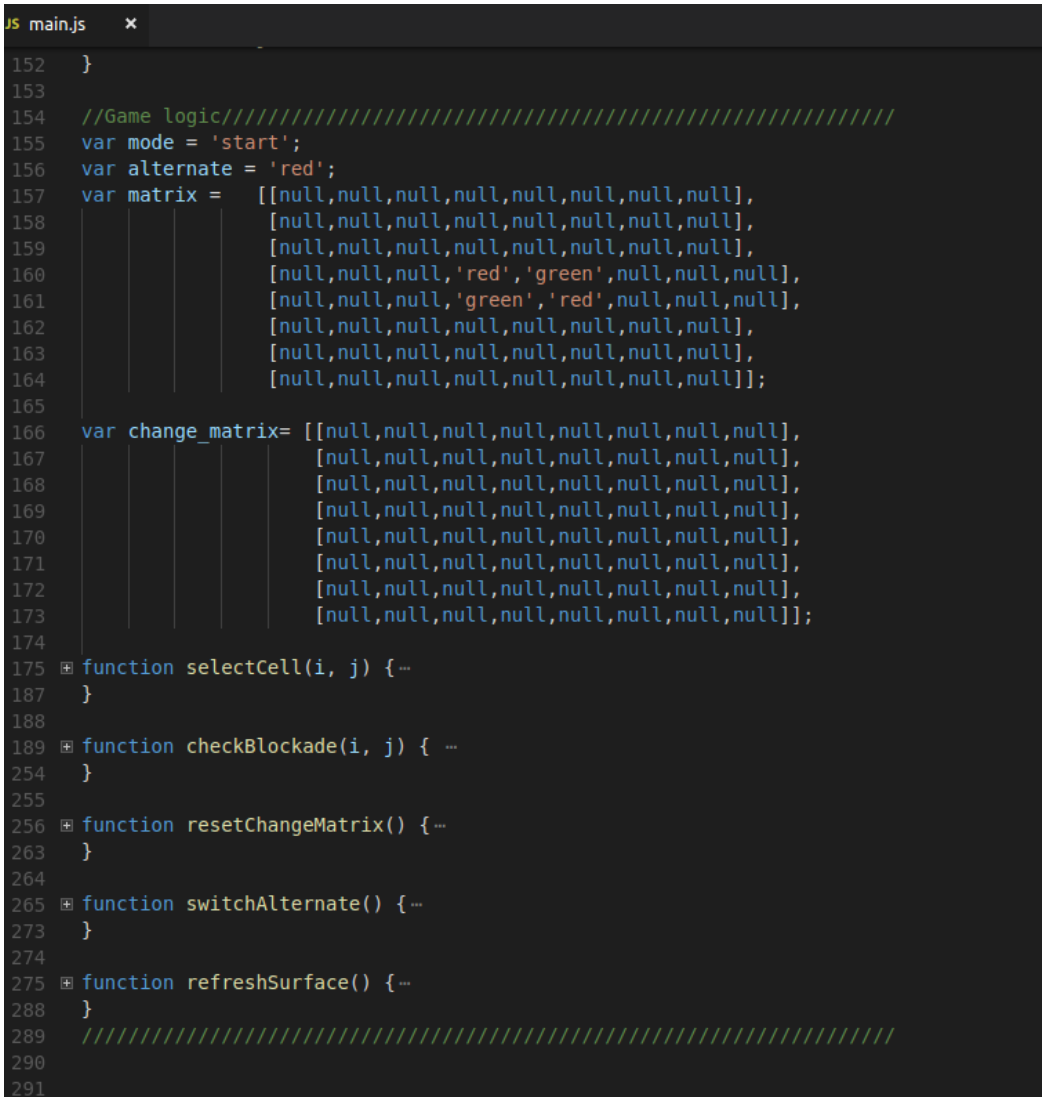
کد های برنامه درون فایل js/main.js قرار دارند.

درون این فایل **ابتدا** متغیر ها و توابع مربوط به initialization برنامه قرار دارند.

توضـــیح اینکه آرایه دو بعدی balls مهره های هر خانه را در خود نگهداری می کند و آرایه دو بعدی godi فرو رفتگی های درون صـــفحه بازی را در خود جای می دهد. علت نگهداری این فرورفتگی ها این ا ست که بعدا در منطق بازی می خواهیم کلیک شدن بر روی آنها را متوجه شویم و در آن خانه مهره قرار دهیم.

تابع init اولین تابعی اســـت که از برنامه اجرا می شـــود. و بقیه ی توابع ابتدایی را فراخوانی می کند.
در تابع initCamera یک دوربین perspective سـاخته می شـود و در مکان مناسـب قرار می گیرد.
تابع initLights دو نور با نوع های AmbientLight و DirectionalLight می سازد و در جای مناسب درون صحنه قرار می دهد.
تابع initAudio فایل با صدای دینگ را load می کند. جهت ا ستفاده این صدا زمانی ا ست که بازیکن خانه ای را انتخاب می کند . با پخش این صدا مهره در صفحه قرار می گیرد.
و در نهایت تابع loadObjects سه شیء خروجی گرفته شده از نرم افزار cinema٤d را در صحنه load می کند. تو ضیح بی شتر اینکه فایل godi و ball یکبار load می شوند اما به تعداد ماتریس بازی تکثیر می شوند. همچنین در این تابع ا ست که ریفرنس این اشـیا درون دو آرایه دوبعدی balls و godi قرار داده می شـوند. و نکته آخر اینکه مهره ها پس از load و قرار گرفتن در جای مناسب، visibleشان false می شود تا اینکه بعدا در جریان بازی و زمان مناسب به کاربر نمایش داده شوند.

در **بخش دوم** فایل main.js متغیر ها و توابع مربوط به منطق بازی قرار گرفته اند.



```
main.js
152   }
153
154   //Game logic/////////////////////////////////////////////////////////
155   var mode = 'start';
156   var alternate = 'red';
157   var matrix =   [[null,null,null,null,null,null,null,null],
158                   [null,null,null,null,null,null,null,null],
159                   [null,null,null,null,null,null,null,null],
160                   [null,null,null,'red','green',null,null,null],
161                   [null,null,null,'green','red',null,null,null],
162                   [null,null,null,null,null,null,null,null],
163                   [null,null,null,null,null,null,null,null],
164                   [null,null,null,null,null,null,null,null]];
165
166   var change_matrix= [[null,null,null,null,null,null,null,null],
167                       [null,null,null,null,null,null,null,null],
168                       [null,null,null,null,null,null,null,null],
169                       [null,null,null,null,null,null,null,null],
170                       [null,null,null,null,null,null,null,null],
171                       [null,null,null,null,null,null,null,null],
172                       [null,null,null,null,null,null,null,null],
173                       [null,null,null,null,null,null,null,null]];
174
175   function selectCell(i, j) {…
187   }
188
189   function checkBlockade(i, j) { …
254   }
255
256   function resetChangeMatrix() {…
263   }
264
265   function switchAlternate() {…
273   }
274
275   function refreshSurface() {…
288   }
289   ////////////////////////////////////////////////////////////////////////
290
291
```

متغیر mode، مد فعلی بازی را مشخص می کند که می تواند شامل سه حالت زیر باشد:

۱- start: زمانی که برنامه اجرا می شود و باید در این مد انیمیشن ابتدایی بازی به کاربر نمایش داده شود.

۲- normal: زمانی اســت که صــفحه بازی آماده اســت تا بازیکن یک خانه را انتخاب نماید.

۳- animation: زمانی اســت که کاربر یک خانه را انتخاب کرده اســت و برنامه در حال اجرای انیمیشن برگرداندن مهره های میانی می باشد.

متغیر alternate نوبت بازیکن ها را نگهداری می کند.

آرایه matrix وضــعیت هر خانه را نگهداری می کند و ســلول های آن می توانند یکی از مقادیر: red ،null و green را داشــته باشــند. بعدا این ماتریس درون صــفحه نمایش داده می شود.

آرایه change_matrix زمانی استفاده می شود که یک بازیکن خانه ای را انتخاب کرده اســت. هنگامی که بازی محاســبه می کند چه مهره هایی باید برگردانده شــوند، آنها را درون این آرایه نگهداری می کند.

تابع selectCell زمانی فراخوانی می شــود که بازیکن روی یک خانه کلیک می کند و مکان آن خانه به این تابع پاس داده می شود. در صورتی که برنامه در mode نرمال قرار داشته باشد، این تابع صدای دینگ را پلی می کند، بر اساس اینکه نوبت چه کسی بوده آرایه ماتریکس را به روز ر سانی میکند، تابع چک کردن محا صره(checkBlockade) را فراخانی می کند و درنهایت صفحه را به روز رسانی می کند.

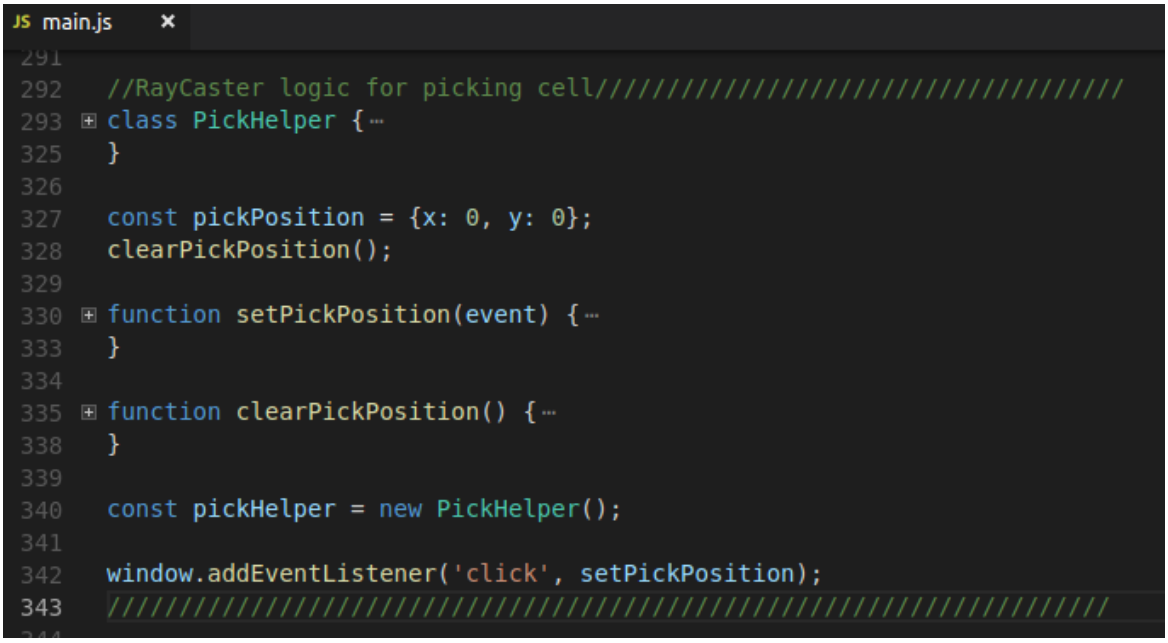
تابع checkBlockade با گرفتن یک خانه، بررســی می کند که آیا با شــروع از آن خانه، مهره های حریف توسط آن بازیکن محاصره شده است یا خیر. مهره های محاصره شده را در آرایه change_matrix مشخص می کند.

تابع resetChangeMatrix تمام خانه های change_matrix را null می کند.

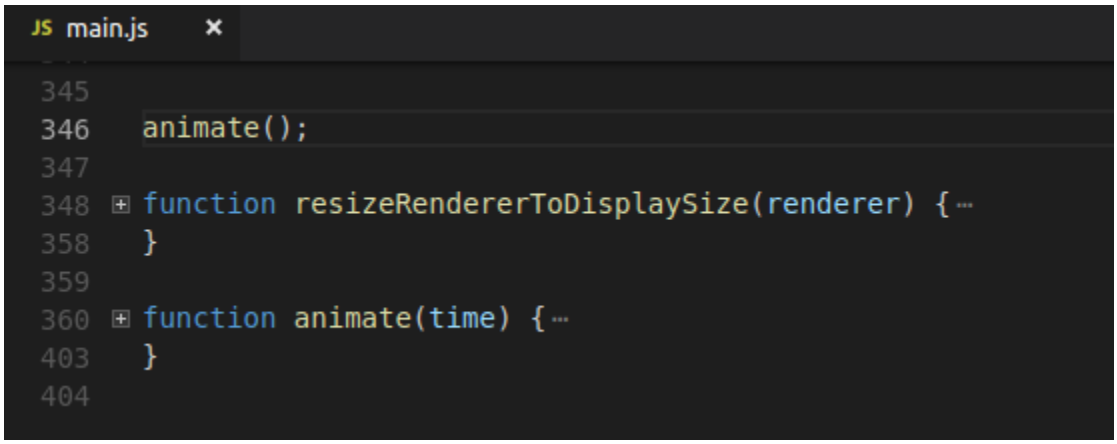
تابع switchAlternate نوبت بازیکن را جابجا می کند و همچنین رنگ مربع داخل هدر را تغییر می دهد.

تابع refreshSurface برا ساس آرایه matrix، مهره های داخل صحنه را به روز ر سانی می کند.

در بخش سوم فایل توابع و کلاس مربوط به Ray Casting جهت تشخیص خانه ای که بازیکن روی آن کلیک کرده است قرار دارند و توضیح خاصی ندارد. تنها نکته این است که پس از تشخیص خانه مورد نظر، تابع selectCell از بخش مربوط به منطق بازی صدا زده می شود.



و در نهایت و **بخش چهارم** فایل، توابع مربوط به انیمیشن قرار گرفته اند.



تابع resizeRendererToDisplaySize زمانی فراخوانی می شـــود که کاربر صـــفحه مرورگر خود را تغییر سایز بدهد که آنگاه باید سایز صحنه مجددا با سایز آن تنظیم شود. در تابع animate بررسی تغییر سایز صفحه انجام می شود. اگر در مد انیمیشن با شیم انیمیشن مربوط به چرخش مهره ها اجرا می شود و اگر در مد start با شیم، انیمیشن مربوط به ابتدای بازی اجرا می شـــود. همچنین فراخوانی Ray Cast از داخل این تابع انجام می شود.