



テトリス・プログラミング

もんだい Stage 2 – Stage 6



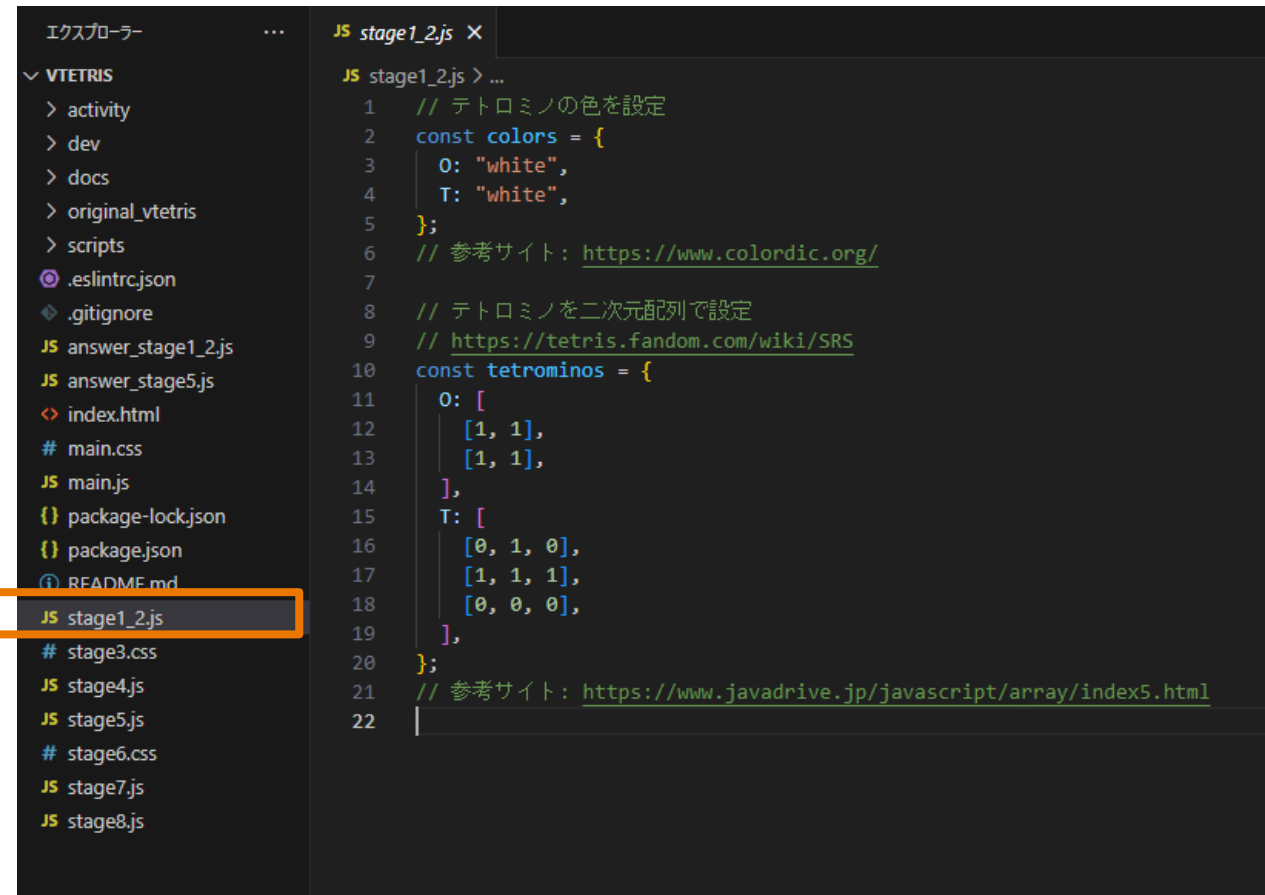
Stage 2

ブロックをふやそう

stage 2 – ブロックの形を作ってみよう ✨

Stage 2でつかうソースコード

このStageでは『Stage1_2.js』をつかうよ！



```
エクスプローラー    ...    JS stage1_2.js X
VTETRIS
  > activity
  > dev
  > docs
  > original_vtetris
  > scripts
  .eslintrc.json
  .gitignore
  JS answer_stage1_2.js
  JS answer_stage5.js
  <> index.html
  # main.css
  JS main.js
  {} package-lock.json
  {} package.json
  @ README.md
  JS stage1_2.js
  # stage3.css
  JS stage4.js
  JS stage5.js
  # stage6.css
  JS stage7.js
  JS stage8.js

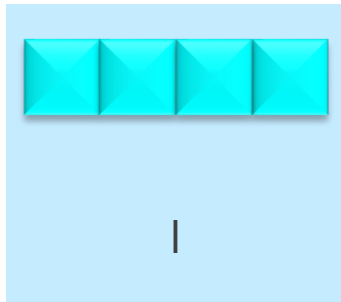
JS stage1_2.js > ...
1  // テトロミノの色を設定
2  const colors = {
3    O: "white",
4    T: "white",
5  };
6  // 参考サイト: https://www.colordic.org/
7
8  // テトロミノを二次元配列で設定
9  // https://tetris.fandom.com/wiki/SRS
10 const tetrominos = {
11   O: [
12     [1, 1],
13     [1, 1],
14   ],
15   T: [
16     [0, 1, 0],
17     [1, 1, 1],
18     [0, 0, 0],
19   ],
20 };
21 // 参考サイト: https://www.javadrive.jp/javascript/array/index5.html
22
```

【問題1】

O や T の書き方を参考に、
他のブロックをどう書くかの問題をやってみよう 😊

問題1：“I” はコードで書くとどうなる？

①.



```
I: [  
  [0, 0, 0, 0],  
  [0, 0, 0, 0],  
  [1, 1, 1, 1],  
],
```

②.

```
I: [  
  [0, 0, 0, 0],  
  [0, 0, 0, 0],  
  [0, 1, 1, 1],  
  [0, 0, 0, 0],  
],
```

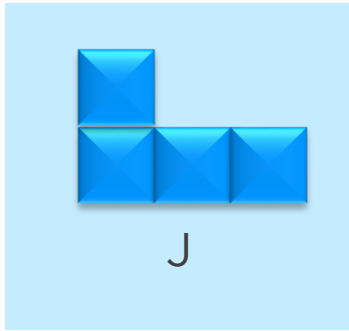
③.

```
I: [  
  [0, 0, 0, 0],  
  [0, 0, 0, 0],  
  [1, 1, 1, 1],  
  [0, 0, 0, 0],  
],
```

【問題2】

O や T の書き方を^{さんこう}参考に、
他のブロックをどう書くかの問題をやってみよう 😊

問題2：“J” はコードで書くとどうなる？



①.

```
J: [  
  [0, 0, 0],  
  [1, 1, 1],  
  [1, 0, 0],  
],
```

②.

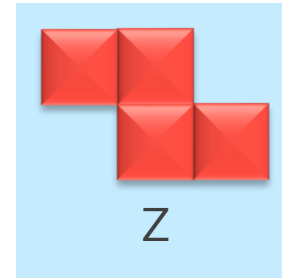
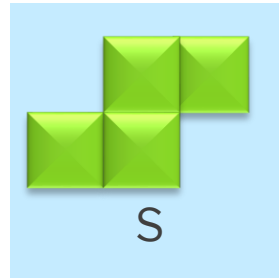
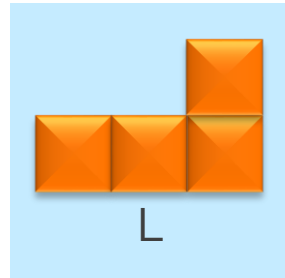
```
J: [  
  [1, 0, 0],  
  [1, 1, 1],  
  [0, 0, 0],  
],
```

③.

```
J: [  
  [1, 0, 0],  
  [1, 1, 1],  
],
```

【問題3】

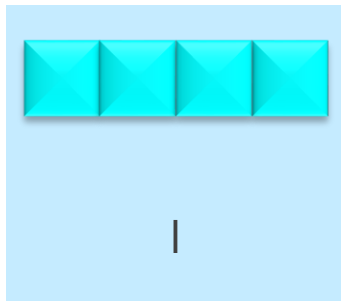
さんこう
O や T の書き方を参考に、
他のブロックをどう書くかの問題をやってみよう 😊
問題3：ほかのかたちはコードで書くとどうなる？



問題1のこたえ

問題1：Iはコードで書くとどうなる？

こたえは③



①.

```
I: [  
  [0, 0, 0, 0],  
  [0, 0, 0, 0],  
  [1, 1, 1, 1],  
],
```

②.

```
I: [  
  [0, 0, 0, 0],  
  [0, 0, 0, 0],  
  [0, 1, 1, 1],  
  [0, 0, 0, 0],  
],
```

③.

```
I: [  
  [0, 0, 0, 0],  
  [0, 0, 0, 0],  
  [1, 1, 1, 1],  
  [0, 0, 0, 0],  
],
```

1がブロックの形を
なぞるように横に4
つならんでいる！

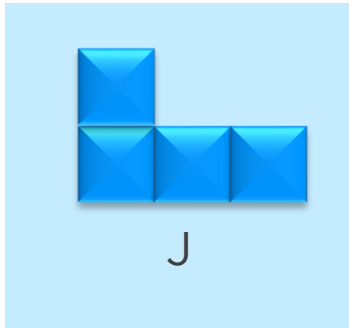
いちばん長いところ
に合わせてタテ
ヨコ同じマス数に
なっている！

1が3れつ目なのはなんで？
⇒ほかの列でもだめじゃないけど、
3れつ目だとあとできれいに
回転するよ！

問題2のこたえ

問題2：Jはコードで書くとどうなる？

こたえは②



1がブロックの形を
なぞるように正し
くならんでいる！

```
J: [  
  [0, 0, 0],  
  [1, 1, 1],  
  [1, 0, 0],  
],
```

②.

```
J: [  
  [1, 0, 0],  
  [1, 1, 1],  
  [0, 0, 0],  
],
```

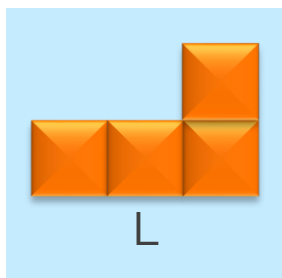
いちばん長いとこ
ろに合わせてタテ
ヨコ同じマス数に
なっている！

③.

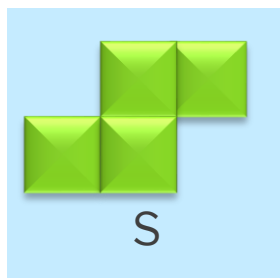
```
J: [  
  [1, 0, 0],  
  [1, 1, 1],  
],
```


問題3のこたえ

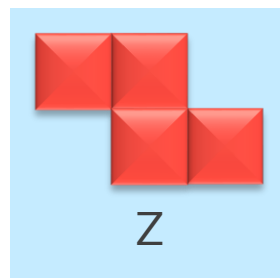
ほかのかたちの書き方も見よう！



```
L: [  
  [0, 0, 1],  
  [1, 1, 1],  
  [0, 0, 0],  
],
```



```
S: [  
  [0, 1, 1],  
  [1, 1, 0],  
  [0, 0, 0],  
],
```



```
Z: [  
  [1, 1, 0],  
  [0, 1, 1],  
  [0, 0, 0],  
],
```

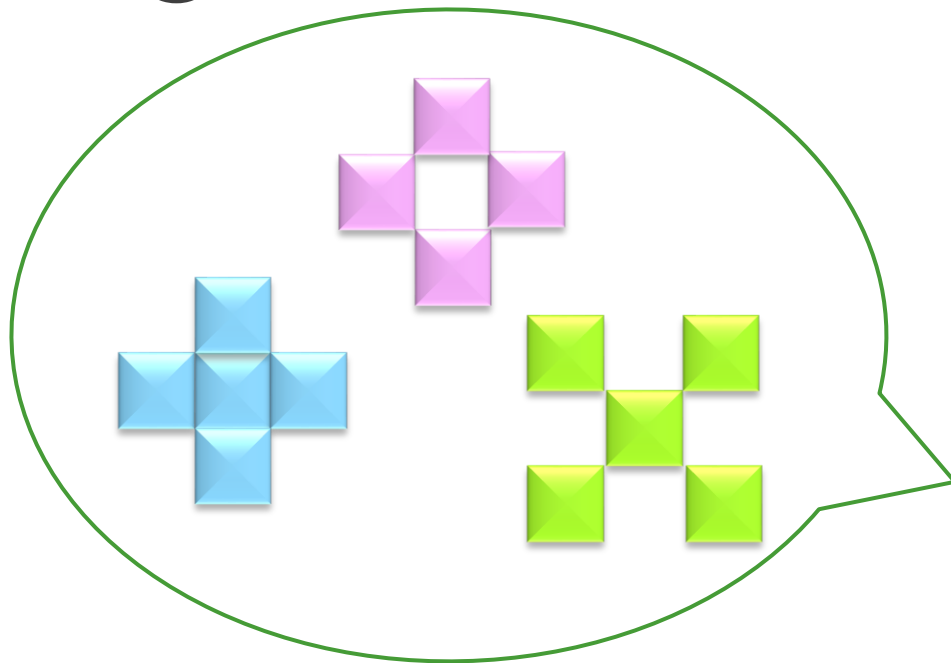
💡 ポイントおさらい 💡

- ✓ ブロックの形をなぞるように1が書かれている！空きマスは0
- ✓ いちばん長いところに合わせてタテヨコ同じマス数になっている

形をアレンジしてもオッケー👉

ブロックをおもしろい形にしてもよし！

コードの書き方のポイントをおさえながら、オリジナルの形もつくってみてね😊





Stage 3

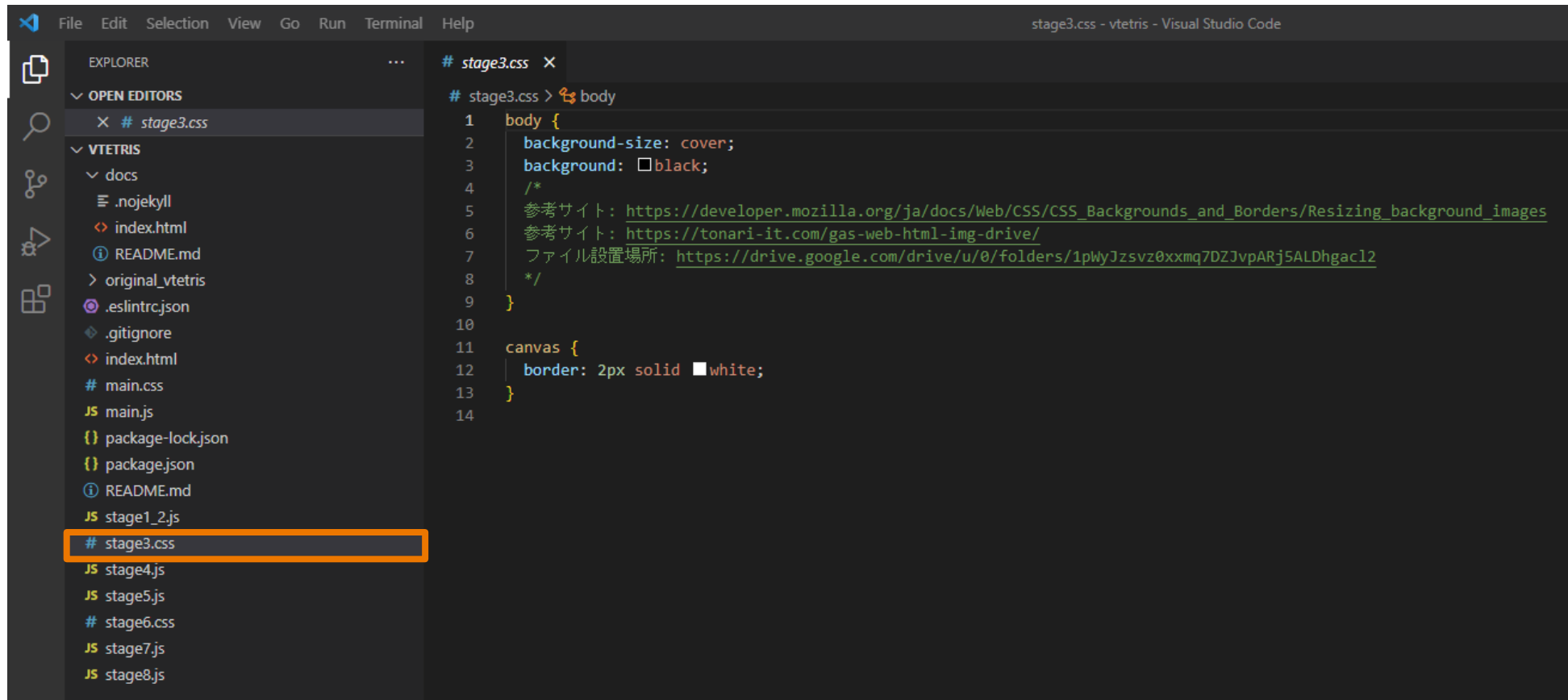
はいけい

背景をアレンジしてみよう

stage 3 – はいけい 背景の色をかえてみよう

Stage 3でつかうソースコード

このStageでは『stage3.css』をつかうよ！



The screenshot shows the Visual Studio Code interface. On the left, the Explorer sidebar displays a project structure for 'VTETRIS'. The file 'stage3.css' is highlighted with an orange rectangle. The main editor area shows the content of 'stage3.css' with the following code:

```
# stage3.css > body
1  body {
2    background-size: cover;
3    background: black;
4    /*
5     参考サイト: https://developer.mozilla.org/ja/docs/Web/CSS/CSS\_Backgrounds\_and\_Borders/Resizing\_background\_images
6     参考サイト: https://tonari-it.com/gas-web-html-img-drive/
7     ファイル設置場所: https://drive.google.com/drive/u/0/folders/1pWyJzsvz0xxmq7DZJvpARj5ALDhgac12
8     */
9  }
10
11  canvas {
12    border: 2px solid white;
13  }
14
```

【問題】

はいけい
背景をアレンジしてみよう

Q. CSS を^{しゅうせい}修正して、^{はいけい}背景を好きな色にかえてみよう！

```
# stage3.css ×
# stage3.css > body
1  body {
2      background-size: cover;
3      background: □black;
4      /*
5       参考サイト: https://developer.mozilla.org/ja/docs/Web/CSS/CSS\_Backgrounds\_and\_Borders/Resizing\_background\_images
6       参考サイト: https://tonari-it.com/gas-web-html-img-drive/
7       ファイル設置場所: https://drive.google.com/drive/u/0/folders/1pWyJzsvz0xxmq7DZJvpARj5ALDhgac12
8       */
9  }
10
11  canvas {
12      border: 2px solid ■white;
13  }
14
```

【問題 こたえ】

はいけい
Q.背景を好きな色にかえてみよう！

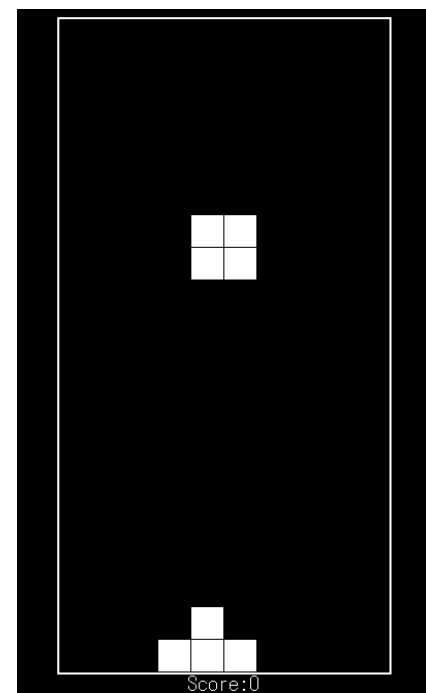
A.“background” を好きな色にかえる！

black を pink に

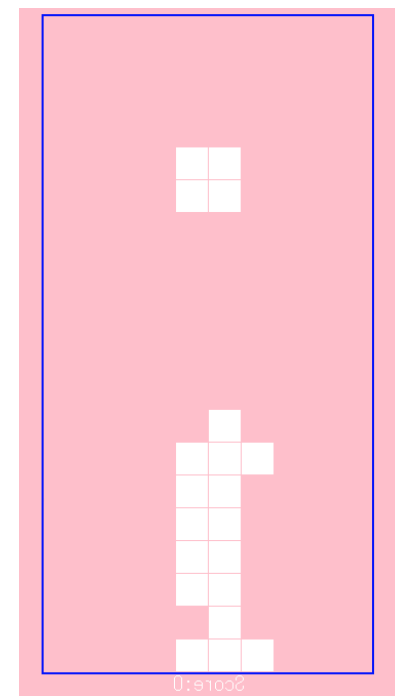
```
# stage3.css > ...
1  body {
2    background-size: cover;
3    background: ■pink;
4    /*
5     参考サイト: https://developer.mozilla.org/ja/
6     参考サイト: https://tonari-it.com/gas-web-htm
7     ファイル設置場所: https://drive.google.com/dr
8     */
9  }
10
11 canvas {
12   border: 2px solid ■blue;
13 }
14
```

こっちをかえると
どうなるかな？

はじめ



こたえ



はいけい

"background" が背景を決めていて、
"border" が わく の色を決めていました₁₆

【チャレンジ問題】

Q.CSS を書いたら画像^{がぞう}を背景^{はいけい}にできそうか？

water.jpg^{はいけい} を背景にする。「？」の中はどう書いたらいいか？

```
body {  
  background-size: cover;  
  background: pink;  
  background-image:url( ? ); <!-- この行を追加してね！ -->  
}  
  
canvas {  
  border: 2px solid blue;  
}
```

water.jpg

https://drive.google.com/uc?id=10UbgMWWGjDd3wCeTu07Ad5-eY-_tbrk7

img/water.png

<https://drive.google.com/water.jpg>

【チャレンジ問題 こたえ】

Q. どう書いたら^{がぞう}画像を^{はいけい}背景にできそうか？

今回は^{がぞう アイディー}https://drive.google.com/uc?id=<画像のID>で
Google Drive上にアップロードした^{がぞう}画像を読みこむことができます

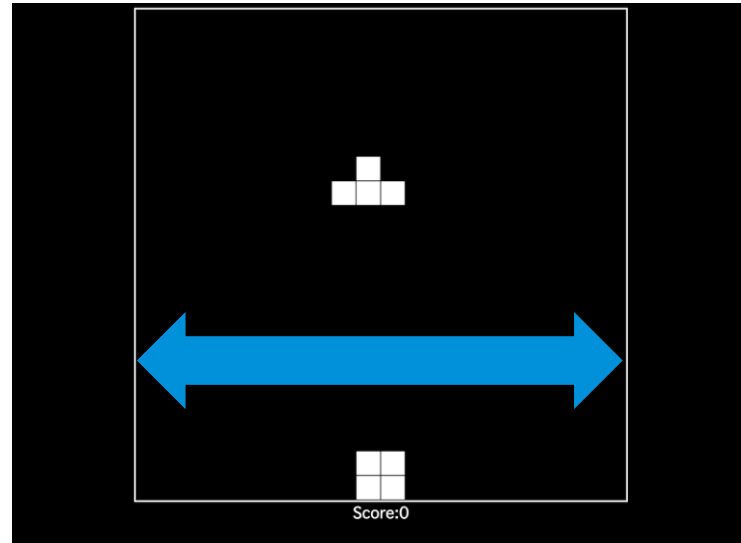
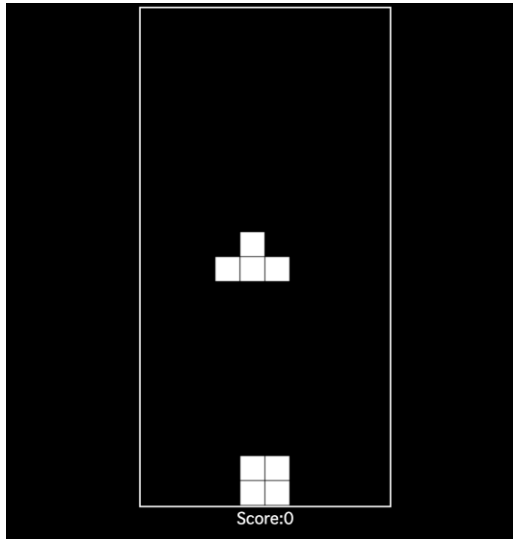
```
body {  
  background-size: cover;  
  background: pink;  
  background-image: url(https://drive.google.com/uc?id=10UbgMWWGjDd3wCeTu07Ad5-eY-_tbrk7);  
}  
  
canvas {  
  border: 2px solid blue;  
}
```




Stage 4

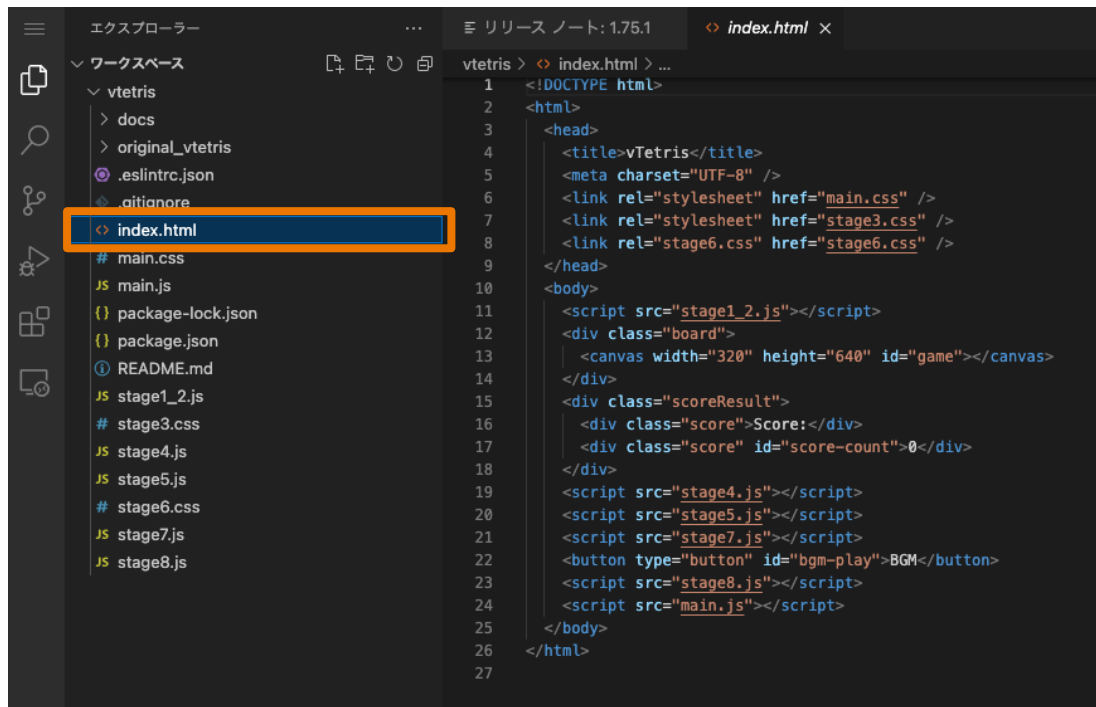
プレイフィールドを広げてみよう

かくちょう
stage 4 – プレイフィールド拡張

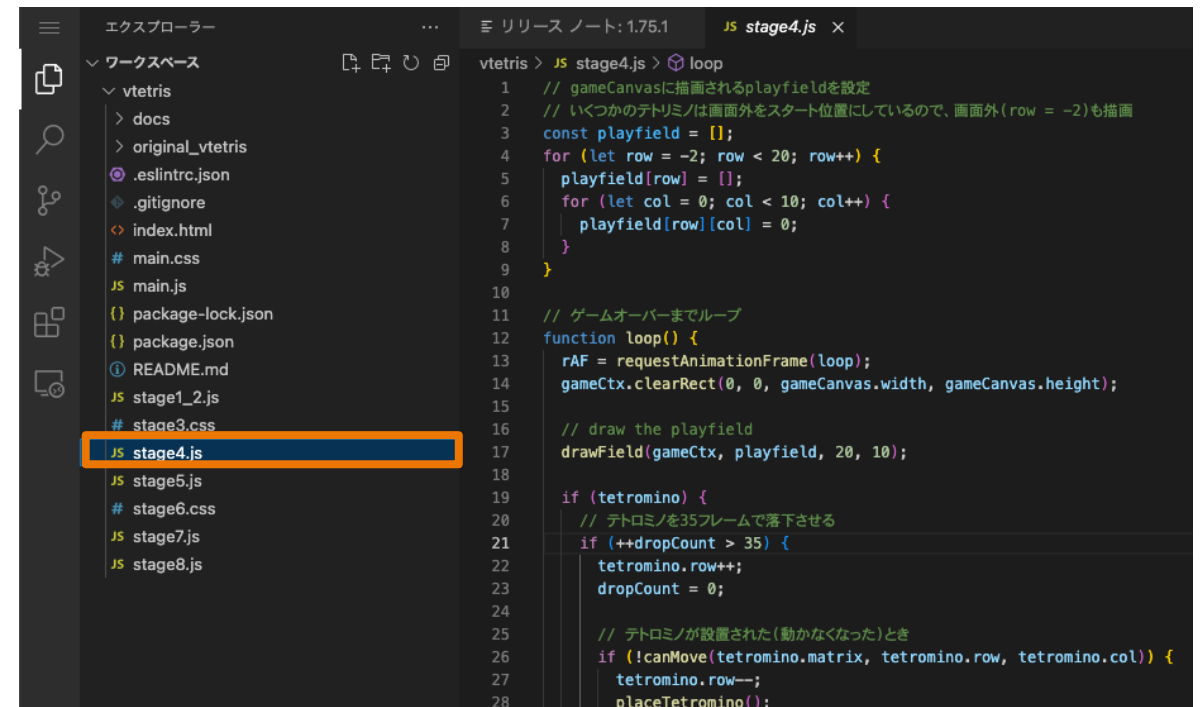


Stage 4でつかうソースコード

このStageでは『index.html』と『Stage4.js』をつかうよ！



The screenshot shows the VS Code interface. On the left, the Explorer sidebar displays the project structure for 'vtetris'. The file 'index.html' is highlighted with an orange box. The main editor shows the content of 'index.html', which includes a DOCTYPE declaration, HTML head with title and meta tags, and a body with a canvas element and several script tags for 'stage1_2.js', 'stage4.js', 'stage5.js', 'stage7.js', 'stage8.js', and 'main.js'.



The screenshot shows the VS Code interface. On the left, the Explorer sidebar displays the project structure for 'vtetris'. The file 'js stage4.js' is highlighted with an orange box. The main editor shows the content of 'js stage4.js', which includes a loop function for drawing the playfield and a tetromino movement function.

【もんだい】プレイフィールドの 横はばを2倍にしてみよう その1

たいしょう

対象のファイルは『index.html』です それぞれどんな数字にかえればいいかな？

```
<!-- プレイフィールドの横はばを2倍にしてみよう -->  
<div class="board">  
  <canvas width="①" height="①" id="game"></canvas>  
</div>
```

①Canvasの大きさは？（元の大きさはwidth="320" height="640"）

width="320" height="1280"

width="320" height="320"

width="640" height="640"

width="320" height="640"

【もんだい】プレイフィールドの横はばを2倍にしてみよう その2

たいしょう

対象のファイルは『Stage4.js』です それぞれどんな数字にかえればいいかな？

```
// gameCanvasに描画されるplayfieldを設定 びょうが せってい  
// いくつかのテトリミノは画面外をスタート位置にしている いちので、画面外(row = -2)も描画 びょうが  
const playfield = [];  
for (let row = -2; ②; row++) {  
  playfield[row] = [];  
  for (let col = 0; ②; col++) {  
    playfield[row][col] = 0;  
  }  
}
```

しより

②行と列の処理をどこまで繰り返す？ (元はrow<20, col<10)

row<20, col<20

row<40, col<20

row<10, col<10

row<40, col<40

【もんだい】プレイフィールドの 横はばを2倍にしてみよう その3

たいしょう

対象のファイルは『Stage4.js』です それぞれどんな数字にかえればいいかな？

```
// ゲームオーバーまでループ
function loop() {
  rAF = requestAnimationFrame(loop);
  gameCtx.clearRect(0, 0, gameCanvas.width, gameCanvas.height);

  // draw the playfield
  drawField(gameCtx, playfield, ③, ③);
```

えが

③遊べるエリアの中で、どの部分を画面に描く？ (元の数字は20, 10)

10, 10

20, 20

200, 100

200, 200

【こたえ】プレイフィールドの 横はばを2倍にしてみよう その1

たいしょう

対象のファイルは 『index.html』 です

```
<!-- プレイフィールドの横はばを2倍にしてみよう -->
<div class="board">
  <canvas width="640" height="640" id="game"></canvas>
</div>
```

①Canvasの大きさは？ (元の大きさはwidth="320" height="640")

width="320" height="1280"

width="320" height="320"

width="640" height="640"

width="320" height="640"

【こたえ】プレイフィールドの横はばを2倍にしてみよう その2

たいしょう

対象のファイルは 『Stage4.js』 です

```
// gameCanvasに描画されるplayfieldを設定 びょうが せってい  
// いくつかのテトリミノは画面外をスタート位置 いちにしている びょうがので、画面外(row = -2)も描画  
const playfield = [];  
for (let row = -2; row < 20; row++) {  
  playfield[row] = [];  
  for (let col = 0; col < 20; col++) {  
    playfield[row][col] = 0;  
  }  
}
```

しより

②行と列の処理をどこまで繰り返す？ (元はrow<20, col<10)

row<20, col<20

row<40, col<20

row<10, col<10

row<40, col<40

【こたえ】プレイフィールドの 横はばを2倍にしてみよう その3

たいしょう

対象のファイルは 『Stage4.js』 です

```
// ゲームオーバーまでループ
function loop() {
  rAF = requestAnimationFrame(loop);
  gameCtx.clearRect(0, 0, gameCanvas.width, gameCanvas.height);

  // draw the playfield
  drawField(gameCtx, playfield, 20, 20);
}
```

えが

③遊べるエリアの中で、どの部分を画面に描く？ (元の数字は20, 10)

10, 10

20, 20

200, 100

200, 200



上級編！



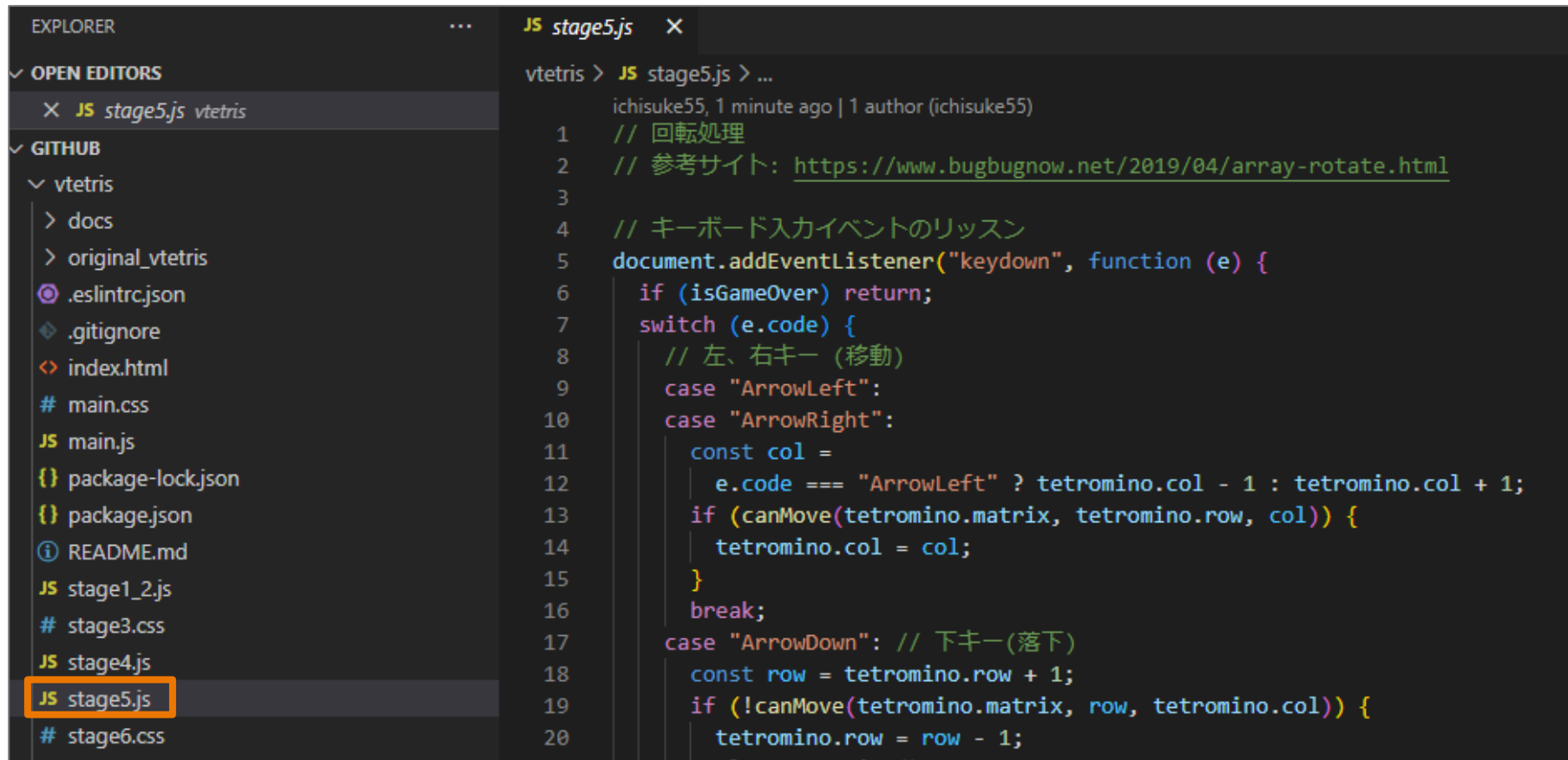
テトロミノを回転させよう



stage 5 – 回転

Stage 5でつかうソースコード

このStageでは『stage5.js』をつかうよ！



The screenshot shows the VS Code interface. On the left, the Explorer sidebar displays the file structure of a project named 'vtetris'. The file 'stage5.js' is highlighted with an orange box. The main editor area shows the content of 'stage5.js', which is a JavaScript file for handling Tetris game logic. The code includes comments in Japanese and JavaScript code for keyboard event listeners and game state management.

```
EXPLORER
...
OPEN EDITORS
  JS stage5.js vtetris
GITHUB
  vtetris
    docs
    original_vtetris
    .eslintrc.json
    .gitignore
    index.html
    # main.css
    JS main.js
    {} package-lock.json
    {} package.json
    README.md
    JS stage1_2.js
    # stage3.css
    JS stage4.js
    JS stage5.js
    # stage6.css

JS stage5.js
vtetris > JS stage5.js > ...
ichisuke55, 1 minute ago | 1 author (ichisuke55)
1 // 回転処理
2 // 参考サイト: https://www.bugbugnow.net/2019/04/array-rotate.html
3
4 // キーボード入力イベントのリッスン
5 document.addEventListener("keydown", function (e) {
6   if (isGameOver) return;
7   switch (e.code) {
8     // 左、右キー (移動)
9     case "ArrowLeft":
10    case "ArrowRight":
11      const col =
12        e.code === "ArrowLeft" ? tetromino.col - 1 : tetromino.col + 1;
13      if (canMove(tetromino.matrix, tetromino.row, col)) {
14        tetromino.col = col;
15      }
16      break;
17    case "ArrowDown": // 下キー(落下)
18      const row = tetromino.row + 1;
19      if (!canMove(tetromino.matrix, row, tetromino.col)) {
20        tetromino.row = row - 1;
```

【もんだい1】回転の関数を作ろう

stage5.js の「^{かいてんしより}// 回転処理」の部分に^{かんすう}回転の関数をかいてみよう

```
JS stage5.js ×
JS stage5.js > ...
1 // 回転処理
2 // 参考サイト: https://www.bugbugnow.net/2019/04/array-rotate.html
3
4 // キーボード入力イベントのリッスン
5 document.addEventListener("keydown", function (e) {
6     if (isGameOver) return;
7     switch (e.code) {
8         // 左、右キー (移動)
9         case "ArrowLeft":
10        case "ArrowRight": {
11            const col =
12                e.code === "ArrowLeft" ? tetromino.col - 1 : tetromino.col + 1;
13            if (canMove(tetromino.matrix, tetromino.row, col)) {
14                tetromino.col = col;
15            }
16            break;
17        }
18    }
19 }
```

かんすう 【もんだい1】回転の関数を作ろう

②回転の関数 rotateBlock

```
function rotateBlock(blockMatrix) {  
  const N = blockMatrix.⑤ - 1;  
  let result = [];  
  // 行数 (ヨコ) をforで回す  
  for (let i = 0; i < blockMatrix.length; ⑥) {  
    // わたされたブロック行数分の二次元配列 (タテ1, 2, ...を入れる箱) を用意  
    result[i] = [];  
    // 列数 (タテ) をforで回す  
    for (let j = 0; j < blockMatrix[0].length; j++) {  
      // ヨコとタテを入れかえてresultの箱にいれていく  
      result[i][j] = blockMatrix[⑦][i];  
    }  
  }  
  // できたresultの箱を結果として返す  
  return result;  
}
```

かんすう さくせい
//rotateBlock関数の作成
//配列の箱は0からスタートするため、1引いておく
けっか
//結果を入れる配列の箱を作る

⑤配列の長さは何で調べる？

pop

length

canMove

matrix

⑥ iに1を足すときの書き方は？

i + i

+1

i++

+i

⑦ヨコの数とタテにするときの計算は？

j

N

j - N

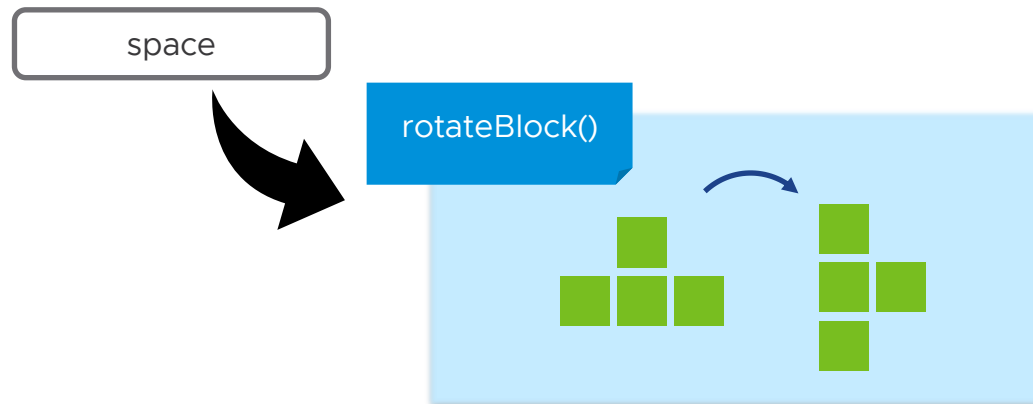
N - j

【もんだい2】キー入力と めりこみ チェック

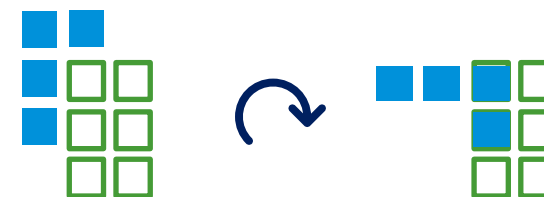
stage5.js にキー入力と めりこみ チェックの処理を^{しより}かいてみよう

```
33     }  
34     break;  
35 }  
36  
37 // 問題: 回転イベントを発生させるkeyEvent条件分岐を追加すること  
38 // 参考サイト: https://developer.mozilla.org/ja/docs/Web/API/KeyboardEvent/keyCode  
39 // 参考サイト: https://www.javadrive.jp/javascript/function/index6.html  
40 }  
41 });  
42
```

Space をおしたら rotateBlock ^{かんすう} 関数で回転



回転したとき、めりこまないよう
チェックを入れる！



【もんだい2】キー入力と めりこみ チェック

①キー入力と③動作チェック

```
case " ① " : // Spaceキー(回転)
// ②が持つ二次元配列データをrotateBlock関数にわたして、結果をmatrixに入れる
const matrix = rotateBlock( ② );
// 回転した後に問題がないかを③関数で確認
if ( ③ (matrix, tetromino.row, tetromino.col)) {
// 動かせる場合は、tetromino.matrixに回転させたあた(④)を代入して、switchをぬける
tetromino.matrix = ④ ;
}
// 回転後のcanMoveでエラー (false) となった場合は、代入をしないでswitchをぬける
break;
```

①Spaceのコード名は？

Delete

KeyS

Space

" "

②ブロックのデータはどこに入っている？

tetrominos

gameCanvas

colors

tetromino.matrix

③動きをチェックする関数は？

canMove

loop

const

placeTetromino

④結果の入った箱は？

tetromino

length

matrix

rotateBlock

【こたえ1】回転の関数を作ろう

②回転の関数 rotateBlock

※緑色はせつめいなので、かかなくてもだいじょうぶ！

```
function rotateBlock(blockMatrix) {
  const N = blockMatrix.length - 1;
  let result = [];
  // 行数 (ヨコ) をforで回す
  for (let i = 0; i < blockMatrix.length; i++) {
    // わたされたブロック行数分の二次元配列 (タテ1, 2, ...を入れる箱) を用意
    result[i] = [];
    // 列数 (タテ) をforで回す
    for (let j = 0; j < blockMatrix[0].length; j++) {
      // ヨコとタテを入れかえてresultの箱にいれていく
      result[i][j] = blockMatrix[N - j][i];
    }
  }
  // できたresultの箱を結果として返す
  return result;
}
```

```
//rotateBlock関数の作成
//配列の箱は0, 1, 2と0からスタートするため、1引いておく
//結果を入れる配列の箱を作る
```

さらにうでをみがくにはアロー関数（かんすう）とmapメソッドを調べてみよう
アロー関数（かんすう）とmapメソッドを使って書くと、こんなに書き方もできるぞ

```
function rotateBlock(blockMatrix) {
  const N = blockMatrix.length - 1;
  const result = blockMatrix.map((row, i) => row.map((col, j) => blockMatrix[N - j][i]));
  return result;
}
```

【こたえ2】キー入力とめりこみチェック

①キー入力と③動作チェック

※緑色はせつめいなので、かかなくてもだいじょうぶ！

```
case "Space": // スペースキー(回転)
    // tetromino.matrixが持つ二次元配列データをrotateBlock関数に渡して、結果をmatrixに入れる
    const matrix = rotateBlock(tetromino.matrix);
    // 回転した後に問題がないかをcanMove関数で確認
    if (canMove(matrix, tetromino.row, tetromino.col)) {
        // 動かせる場合は、tetromino.matrixに回転させたあたい(matrix)を代入して、switchをぬける
        tetromino.matrix = matrix;
    }
    // 回転後のcanMoveでエラー（false）となった場合は、代入をしないでswitchをぬける
    break;
```


どこに書くとよいかな？

カッコ () {} がどこからどこまでか見てみよう 🔭

キー入力とめりこみチェック

```
26 case "ArrowUp": // 上キー(ハードドロップ)
27     while (canMove(tetromino.matrix, tetromino.row + 1, tetromino.col)) {
28         tetromino.row++;
29     }
30     break;
31
32 // 問題: 回転イベントを発生させるkeyEvent条件分岐を追加すること
33 // 参考サイト: https://developer.mozilla.org/ja/docs/Web/API/KeyboardEvent/keyCode
34 // 参考サイト: https://www.javadrive.jp/javascript/function/index6.html
35 }
```

7行目の“switch (e.which) { “から
35行目の “ } ” までがひとかたまり

ほかのcaseにならべて、
30行目にcase 32を書いてあげよう

かんすう 回転の関数

```
1 // 回転処理
2 // 参考サイト: https://www.bugbugnow.net/2019/04/array-rotate.html
3
4 // キーボード入力イベントのリッスン
```

かんすう
回転の関数はキー入力とはべつの機能だ
きのう

4行目に書いてあげよう



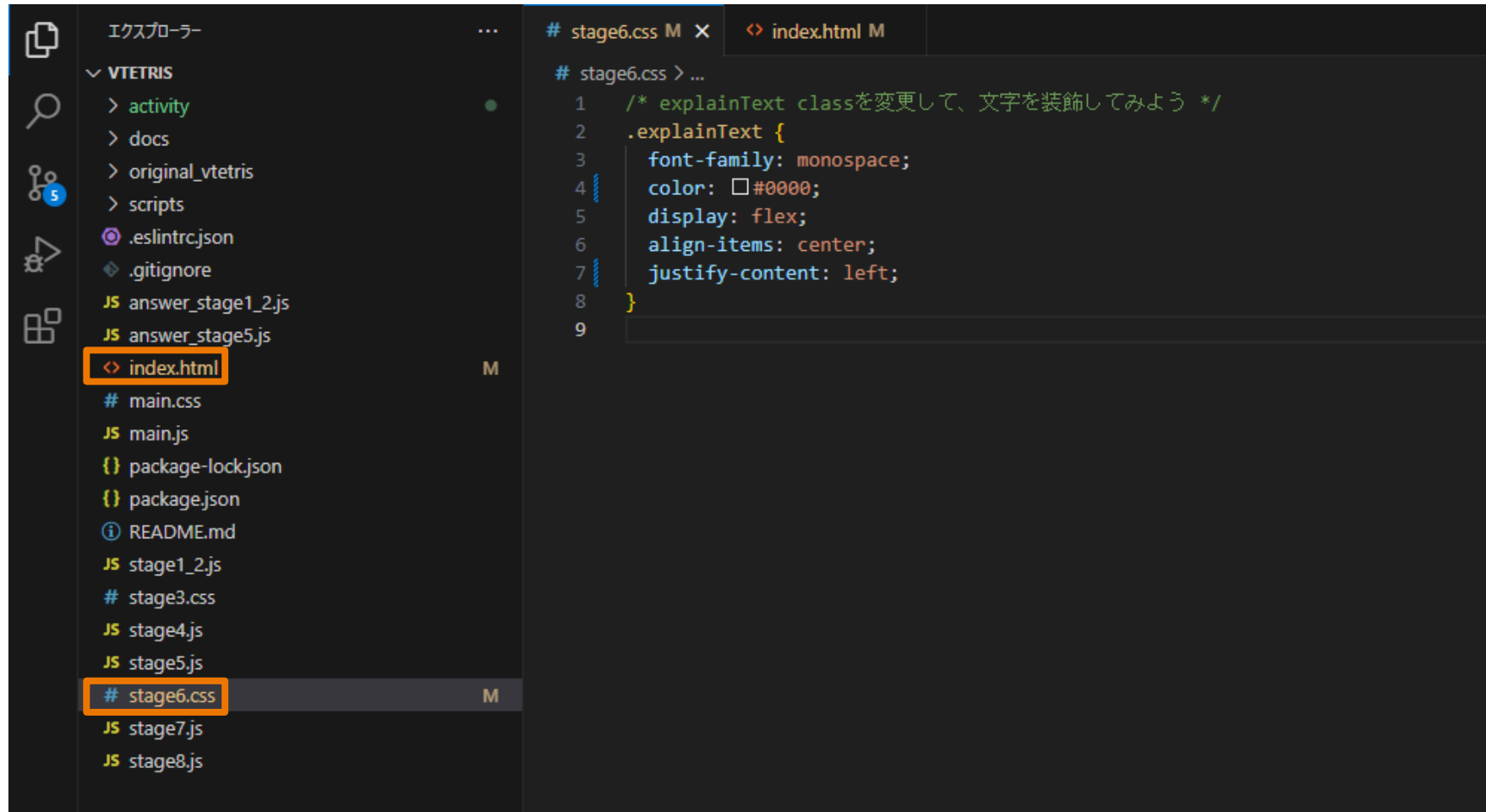
仕上げ！

文字を入れよう！

stage 6 – 名前を入れよう

Stage 6でつかうソースコード

このStageでは『index.html』と『stage6.css』をつかうよ！



```
# stage6.css M X <> index.html M
# stage6.css > ...
1  /* explainText classを変更して、文字を装飾してみよう */
2  .explainText {
3    font-family: monospace;
4    color: #0000;
5    display: flex;
6    align-items: center;
7    justify-content: left;
8  }
9
```

【もんだい】つくった人の名前をいれよう

Index.html

```
<!-- 名前を入れてみよう！タグはexplainTextを使用する -->
<p class = "①">Coded by マイケル</p>
```

①スタイルのclassをよび出すには何をいれる？

style

explainText

.explainText

Stage6.css

```
/* explainText classをへんこう変更して、文字をそうしょく装飾してみよう */
.explainText {
  font-family: monospace;
  color: ②;
  display: flex;
  align-items: center;
  justify-content: ③;
}
```

②文字色を白くするには何をいれる？

white

red

black

green

③文字を真ん中におくには？

left

right

center

end

【こたえ】つくった人の名前をいれよう

Index.html

```
<!-- 名前を入れてみよう！タグはexplainTextを使用する -->
<p class = "explainText">Coded by マイケル</p>
```

Stage6.css

```
/* explainText classを変更して、文字を装飾してみよう */
.explainText {
  font-family: monospace;
  color: white;
  display: flex;
  align-items: center;
  justify-content: center;
}
```

色 (color) などのオプションのことを
「プロパティ (Property)」というよ

プロパティを調べれば、文字の大きさ
(font-size) などもかえられる
ここまでできた人はためしてみよう！！

CSSのプロパティ (font-size) フォント サイズ
<https://developer.mozilla.org/ja/docs/Web/CSS/font-size>

Thank You