**Detailed Process Design Tic tak toe blueprint**

**Author:** Ryan Zhao

**Date:** 2022-03-15

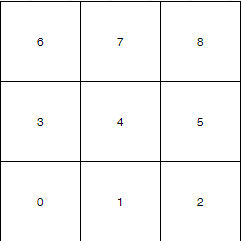
|  |  |  |
| --- | --- | --- |
| **Rev.** | **Changes** | **Date** |
| 01 | Initial draft | 2022-03-15 |
| 02 |  |  |
| 03 |  |  |
| 04 |  |  |
| 05 |  |  |
| 06 |  |  |

**Background**

Great one small game Tic-tak-toe for the testing

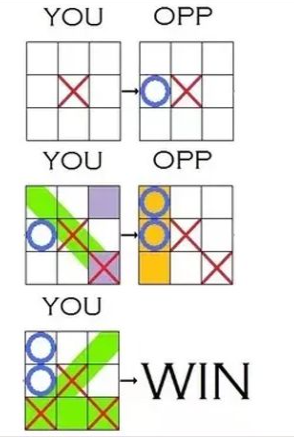
**Flow Chart – Basic Anlysis**

Tic-Tak-Toe game would set one 3\*3 board as a chessboard

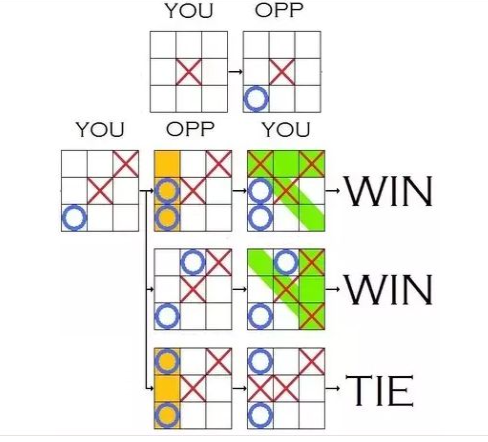


Our propose is that always keep computer win the game. So we should do some analysis.

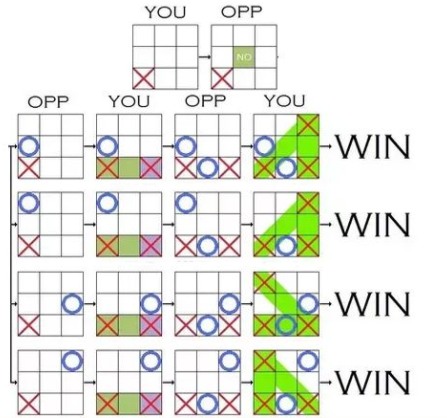
1. Put the chess in position 4.



We set different possibility



1. Put the chess in 0.2.6.8



We will great programming base on below conditions.

1. Keep computer first.
2. Get position 0.2.4.6.8.

**Project detail**

1. **Set basic frame of board.**

Design one board as below

width:160px;

height:160px;

and draw position as below

<div class="main">

<div id="tic-1" class="tic">

<span id="span-1" class="tic-span"></span>

</div>

<div id="tic-2" class="tic">

<span id="span-2" class="tic-span"></span>

</div>

<div id="tic-3" class="tic">

<span id="span-3" class="tic-span"></span>

</div>

<div id="tic-4" class="tic">

<span id="span-4" class="tic-span"></span>

</div>

<div id="tic-5" class="tic">

<span id="span-5" class="tic-span"></span>

</div>

<div id="tic-6" class="tic">

<span id="span-6" class="tic-span"></span>

</div>

<div id="tic-7" class="tic">

<span id="span-7" class="tic-span"></span>

</div>

<div id="tic-8" class="tic">

<span id="span-8" class="tic-span"></span>

</div>

<div id="tic-9" class="tic">

<span id="span-9" class="tic-span"></span>

</div>

</div>

1. **Detail of function**

Set first position which would be set by computer side.

var proStep = [0, 2, 6, 8, 4];

and now computer would choose middle or four conners.

We set two options in programming.

* In the middle

group[proStep[posit]] = 1; //电脑下棋

$("#span-" + (proStep[posit] + 1)).html(cmpt);

judge();

return;

}

if (step === 2) {

//如果是电脑下的第二步,分两种情况，分别是电脑第一步下了正中和四个角

if (group[4] === 1) {

//如果电脑第一步下的正中，又分两种情况，对方下的是四个角还是中间

var corStep = [0, 2, 6, 8]; //四个角在group的索引

for (var t = 0; t < 4; t++) {

if (group[corStep[t]] === 2) {

//如果玩家下的是某一个角，那就下他对角

var posit = 0; //这里表示的是电脑要下的位置

if (corStep[t] === 0) {

posit = 8;

} else if (corStep[t] === 8) {

posit = 0;

} else if (corStep[t] === 2) {

posit = 6;

} else if (corStep[t] === 6) {

posit = 2;

}

posit = parseInt(posit);

group[posit] = 1; //电脑下棋

$("#span-" + (posit + 1)).html(cmpt);

judge();

return;

}

}

//电脑下的不是某个角，而是在每一行或列的中间位置,电脑就下一个靠着它的角

var posit\_g = [0, 0]; //如果下在中间，就会有两个角靠着它，随机选一个

var posit = 0; //这就是随机选择之后的位置

if (group[1] === 2) {

posit\_g[0] = 0;

posit\_g[1] = 2;

} else if (group[3] === 2) {

posit\_g[0] = 0;

posit\_g[1] = 6;

} else if (group[5] === 2) {

posit\_g[0] = 2;

posit\_g[1] = 8;

} else if (group[7] === 2) {

posit\_g[0] = 6;

posit\_g[1] = 8;

}

posit = posit\_g[parseInt(Math.random() \* 2)];

posit = parseInt(posit);

group[posit] = 1; //电脑下棋

$("#span-" + (posit + 1)).html(cmpt);

judge();

return;

* In 4 conners

else {

//如果电脑第一步下的不是正中，而是四个角

//分两种情况，如果对方没下正中，那么就下正中，如果对方下了正中，就下第一步的对角

if (group[4] === 0) {

//玩家没下正中，则下正中

group[4] = 1;

$("#span-5").html(cmpt);

judge();

return;

}

//下第一步的对角

var posit = 0; //记录要下的从0起的位置

if (group[0] === 1) {

posit = 8;

} else if (group[8] === 1) {

posit = 0;

} else if (group[2] === 1) {

posit = 6;

} else if (group[6] === 1) {

posit = 2;

}

posit = parseInt(posit);

group[posit] = 1; //电脑下棋

$("#span-" + (posit + 1)).html(cmpt);

judge();

return;

}