# ~~Code snippet 1: Creating the game~~

1. <script src="Scripts/game/TicTacToeGame.js" type="text/javascript"></script>
   * 1. var game = new TicTacToeGame();

# ~~Code snippet 2: Injecting the game~~

this.game = game;

# ~~Code snippet 3: Determining winner~~

if (this.game.isTie()) {

this.viewModel.isTie(true);

this.viewModel.currentColor(TTTColor.Empty);

return;

}

if (this.game.hasWinner()) {

this.viewModel.winnerColor(this.game.getWinner());

this.viewModel.currentColor(TTTColor.Empty);

return;

}

# ~~Code snippet 4: Validating and making the move~~

1. if (!this.game.isValid(x, y, color))
2. return;
3. this.game.move(x, y, color);

# ~~Code snippet 5: Displaying the game result~~

<div data-bind="visible: isTie()">

Tie!!</div>

<div data-bind="visible: winnerColor() == TTTColor.Cross">

X won!!</div>

<div data-bind="visible: winnerColor() == TTTColor.Circle">

O won!!</div>

# Code snippet 1: Implementing HasWinner

return this.getWinner() != TTTColor.Empty;

* 1. (Code Snippet *- hasWinner*)
  2. JScript
  3. return this.getWinner() != TTTColor.Empty;

# Code snippet 2: Implementing IsTie

if (this.hasWinner())

return false;

for (var x = 0; x < 3; x++)

for (var y = 0; y < 3; y++)

if (this.board[x][y] == TTTColor.Empty)

return false;

return true;

* 1. (Code Snippet *- isTie*)
  2. JScript
  3. if (this.hasWinner())
  4. return false;
  5. for (var x = 0; x < 3; x++)
  6. for (var y = 0; y < 3; y++)
  7. if (this.board[x][y] == TTTColor.Empty)
  8. return false;
  9. return true;

# Code snippet 3: Creating the socket

<script src="Scripts/socket.io.js" type="text/javascript"></script>

var socket = io.connect("http://localhost:8080", {

transports:['xhr-polling']

});

* 1. (Code Snippet – *socket creation*)
  2. JScript
  3. var socket = io.connect("http://localhost:8080", {
  4. transports:['xhr-polling']
  5. });

# Code snippet 4: setting Query strings parameters

var gameId = getQueryVariable("id");

var isFirst = getQueryVariable("first");

function getQueryVariable(variable) {

var query = window.location.search.substring(1);

var vars = query.split("&");

for (var i = 0; i < vars.length; i++) {

var pair = vars[i].split("=");

if (pair[0] == variable) {

return pair[1];

}

}

}

# Code snippet 5: Setting the game id owner

… = new controller(.., gameId);

controller.start(isFirst);

# Code snippet 6: Injecting socket in controller

1. this.socket = socket;
2. this.socket.emit('join', gameId);
3. this.socket.on('command', onNewCommand);
4. function onNewCommand(gameAction) {
5. if (gameAction.Type != 1)
6. return;
8. var x = parseInt(gameAction.CommandData.x);
9. var y = parseInt(gameAction.CommandData.y);
10. var color = gameAction.CommandData.color;
11. if (!controller.game.isValid(x, y, color))
12. return;
13. controller.game.move(x, y, color);
14. controller.board.drawMove(x, y, color);
15. controller.updateGameStatus();
16. }
    1. (Code Snippet – *socket injection*)
    2. JScript
    3. this.socket = socket;
    4. this.socket.emit('join', gameId);
    5. this.socket.on('command', onNewCommand);
    6. function onNewCommand(gameAction) {
    7. if (gameAction.Type != 1)
    8. return;
    10. var x = parseInt(gameAction.CommandData.x);
    11. var y = parseInt(gameAction.CommandData.y);
    12. var color = gameAction.CommandData.color;
    13. if (!controller.game.isValid(x, y, color))
    14. return;
    15. controller.game.move(x, y, color);
    16. controller.board.drawMove(x, y, color);
    17. controller.updateGameStatus();
    18. }

# Code snippet 7: Sending command through socket.IO

1. var action = { Type: 1, CommandData: { x: x, y: y, color: color }};
2. this.socket.emit('command', action);
   1. (Code Snippet – *socket send*)
   2. JScript
   3. var action = { Type: 1, CommandData: { x: x, y: y, color: color }};
   4. this.socket.emit('command', action);

# Code snippet 8: Setting the player

if (isFirst) {

this.viewModel.playerColor(TTTColor.Cross);

}

else {

this.viewModel.playerColor(TTTColor.Circle);

}

**Code snippet 9: Validating player turn in the On move**

if (this.viewModel.playerColor() != this.viewModel.currentColor())

return;