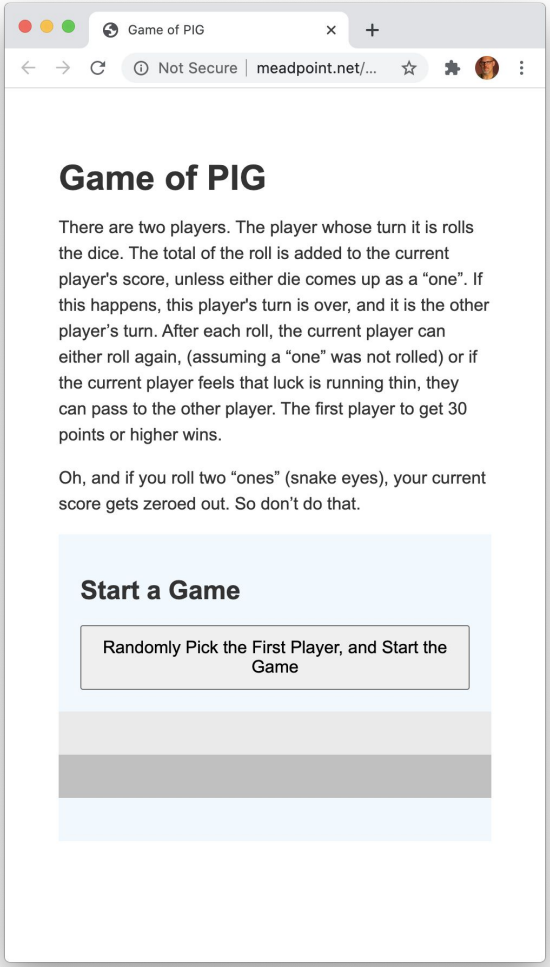
Game of Pig

built in JavaScript

The Basic Game

This is a very basic version with very little styling.



HTML File

The HTML file has five elements you will need to access, based on the state of the game.

```
<div id="gamecontrol">
    <h2>Start a Game</h2>
    <button id="startgame">Randomly
</div>

<div id="game"></div>
<div id="actions"></div>
<div id="score"></div>
```

Add Variables for DOM Elements

Add variables for these DOM elements, so you have quick access to them in your code.

```
<script>

      var startGame = document.getElementById('startgame');
      var gameControl = document.getElementById('gamecontrol');
      var game = document.getElementById('game');
      var score = document.getElementById('score');
      var actionArea = document.getElementById('actions');

</script>
```

Keeping Track of Game Data

You will use an object to keep track of data in the game. Add this object to the script.

```
var gameData = {
      dice: ['1die.jpg', '2die.jpg', '3die.jpg',
               '4die.jpg', '5die.jpg', '6die.jpg'],
      players: ['player 1', 'player 2'],
      score: [0, 0],
      roll1: 0,
      roll2: 0,
      rollSum: 0,
      index: 0,
      gameEnd: 29
};
```

Start Game!

This click handler will start the game and change the content of the gameControl .

```
startGame.addEventListener("click", function(){
    // randomly set game index here...
    gameControl.innerHTML = '<h2>The Game Has Started</h2>';
    gameControl.innerHTML += '<button id="quit">Wanna Quit?</button>';

    document.getElementById('quit').addEventListener("click", function(){
        location.reload();
    });

    console.log("set up the turn!");
});
```

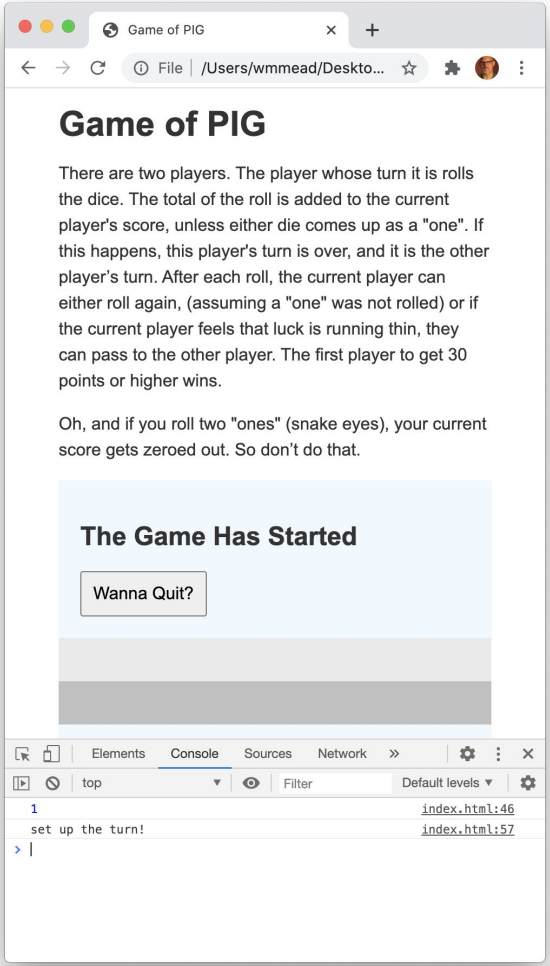
To randomly set the index, replace the comment at the top with this:

```
gameData.index = Math.round(Math.random());
```

Test Game Start!

Test frequently and often! Running this is providing what we expect so far...

Notice that sometimes the gameData.index is 0 and sometimes it's 1



Set Up the Turn

Add the setUpTurn() function and change the console.log in the startGame function to call this function.

Replace the comment with the actual game variable:

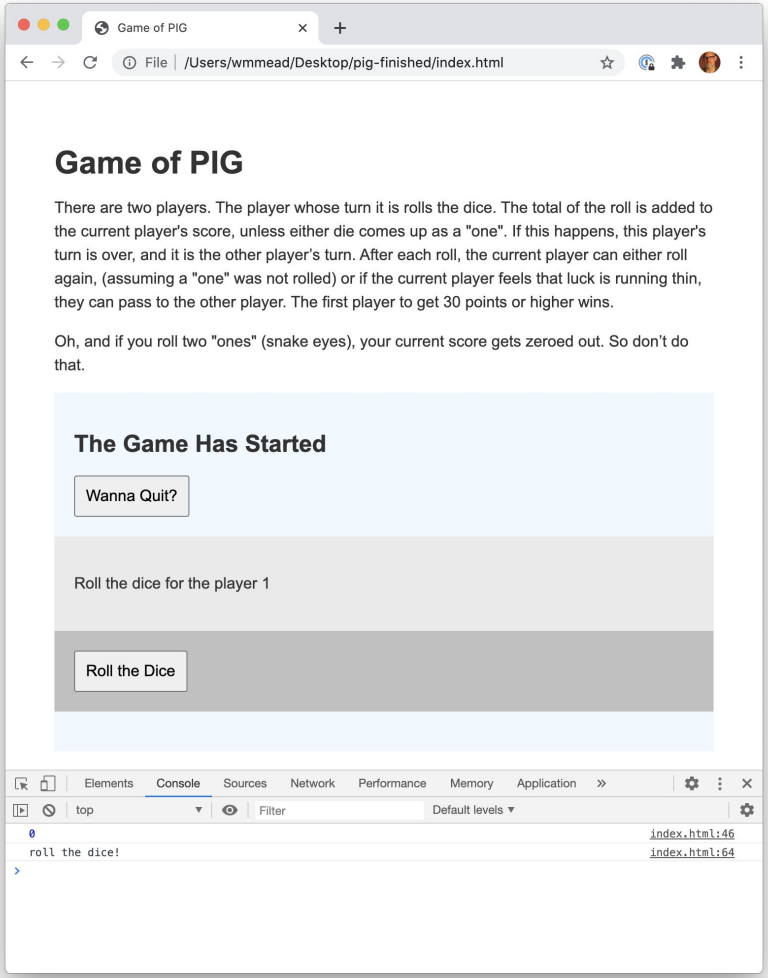
```
gameData.players[gameData.index]
```

```
function setUpTurn() {
    game.innerHTML = `<p>Roll the dice for the ${/*gameData player variable here!*/}</p>`;
    actionArea.innerHTML = '<button id="roll">Roll the Dice</button>';
    document.getElementById('roll').addEventListener('click', function(){

        console.log("roll the dice!");

    });
}
```

Test Set Up Turn

Test the setUpTurn() function. You should be getting a 0 for player one or a 1 for player two.

The roll the dice will only fire when you click the button.



Throwing the Dice

Throwing the dice is the guts of the game, and is the big function. This is just the start of the function.

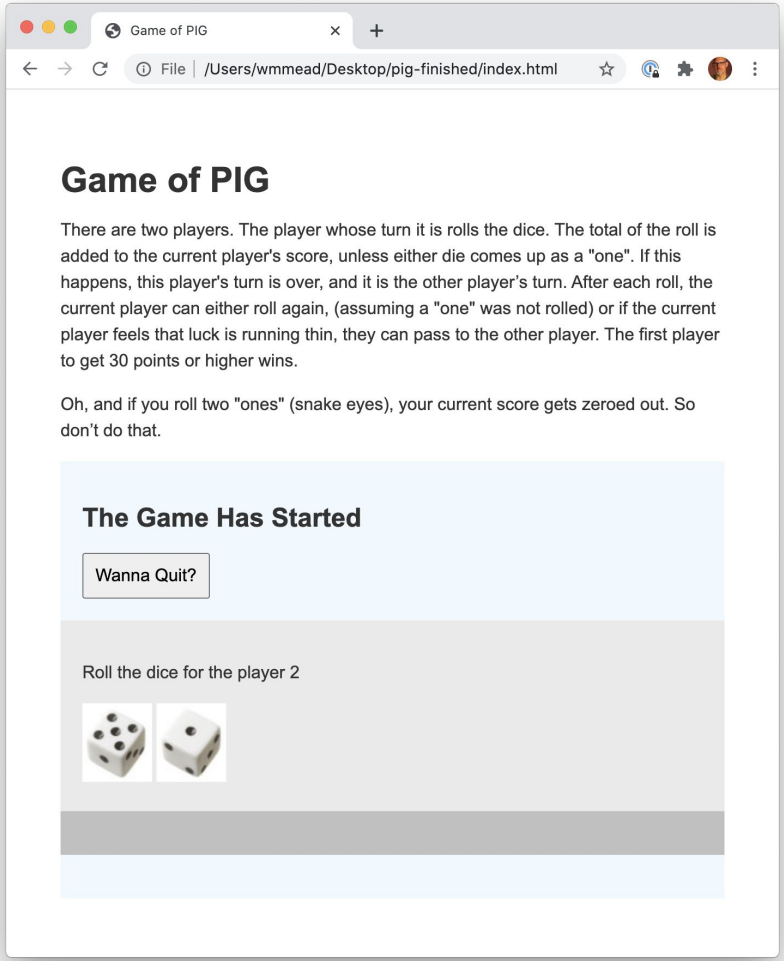
The code below does the steps on the right.

1. Clears out the actionArea
2. Records the two rolls of the dice
3. Sets a message and shows the dice
4. Totals the rolls

```
function throwDice(){
    actionArea.innerHTML = '';
    gameData.roll1 = Math.floor(Math.random() * 6) + 1; //using ceil could result in a zero
    gameData.roll2 = Math.floor(Math.random() * 6) + 1;
    game.innerHTML = `<p>Roll the dice for the ${gameData.players[gameData.index]}</p>`;
    game.innerHTML += `<img src="${gameData.dice[gameData.roll1-1]}">
                       <img src="${gameData.dice[gameData.roll2-1]}">`;
    gameData.rollSum = gameData.roll1 + gameData.roll2;
}
```

Test Throwing

If you replace the call to the console.log() in the setUpTurn() function with a call to the throwDice() function, you should be able to test it and see the dice show up.



Add the Selection Statements

Add these three selection statements into the throwDice() function. They will be used to handle what happens in the game, depending on what comes up on the dice.

```
// if two 1's are rolled...
if( gameData.rollSum === 2 ){
    console.log("snake eyes were rolled");
}
// if either die is a 1...
else if(gameData.roll1 === 1 || gameData.roll2 === 1){
    console.log("one of the two dice was a 1");
}
// if neither die is a 1...
else {
    console.log("the game proceeds");
}
```

Snake Eyes!

Now you fill out the selection statements. The first one handles the case where the sum of the two dice is 2. If that happens, two “ones” came up in the roll.

```
// if two 1's are rolled...
if( gameData.rollSum === 2 ){
    game.innerHTML += '<p>Oh snap! Snake eyes!</p>';
    gameData.score[gameData.index] = 0;
    gameData.index ? (gameData.index = 0) : (gameData.index = 1);
    // Show the current score
    setTimeout(setUpTurn, 2000);
}
```

A 1 Was Rolled

Again we see the ternary operator, which switches the player, then set the message in the game area and set up the next turn (after waiting two seconds).

```
// if either die is a 1...
else if(gameData.roll1 === 1 || gameData.roll2 === 1){
    gameData.index ? (gameData.index = 0) : (gameData.index = 1);
    game.innerHTML += `<p>Sorry, one of your rolls was a one, switching to  ${
        gameData.players[gameData.index]
    }</p>`;
    setTimeout(setUpTurn, 2000);
}
```

The Else Statement

Here is the else statement that runs if no “1” was rolled.

```
// if neither die is a 1...
else {
    gameData.score[gameData.index] = gameData.score[gameData.index] + gameData.rollSum;
    actionArea.innerHTML = '<button id="rollagain">Roll again</button> or <button
    id="pass">Pass</button>';

    document.getElementById('rollagain').addEventListener('click', function () {
            setUpTurn();
        });

    document.getElementById('pass').addEventListener('click', function () {
            gameData.index ? (gameData.index = 0) : (gameData.index = 1);
            setUpTurn();
        });

    //check winning condition!
}
```

Checking for a Win

This code could have just gone inside the throwDice() function, but that function is getting really long already, so it makes sense to break it out into its own function.

Check to see if the player has won. If that is true, Set a message with the final score, clear out the action area of the DOM and change the text inside the quit button to “Start a New Game”.

```
function checkWinningCondition(){
    if(gameData.score[gameData.index] > gameData.gameEnd){
        score.innerHTML = `<h2>${gameData.players[gameData.index]}
        wins with ${gameData.score[gameData.index]} points!</h2>`;

        actionArea.innerHTML = '';
        document.getElementById('quit').innerHTML = "Start a New Game?";
    }
    else{
        // Show current score...
    }
}
```

Showing the Current Score

If the player has not won yet, inside the else statement, you need to update the score area of the board with the current score.

```
else {
    // show current score...
    score.innerHTML = `<p>The score is currently <strong>${gameData.players[0]}
    ${gameData.score[0]}</strong> and <strong>${gameData.players[1]}
    ${gameData.score[1]}</strong></p>`;
}
```

Show the Current Score Function

Here is the showCurrentScore() function. Be sure to call this function in two places:

1. In the checkWinningCondition() function,
2. And in the if statement of the throwDice function

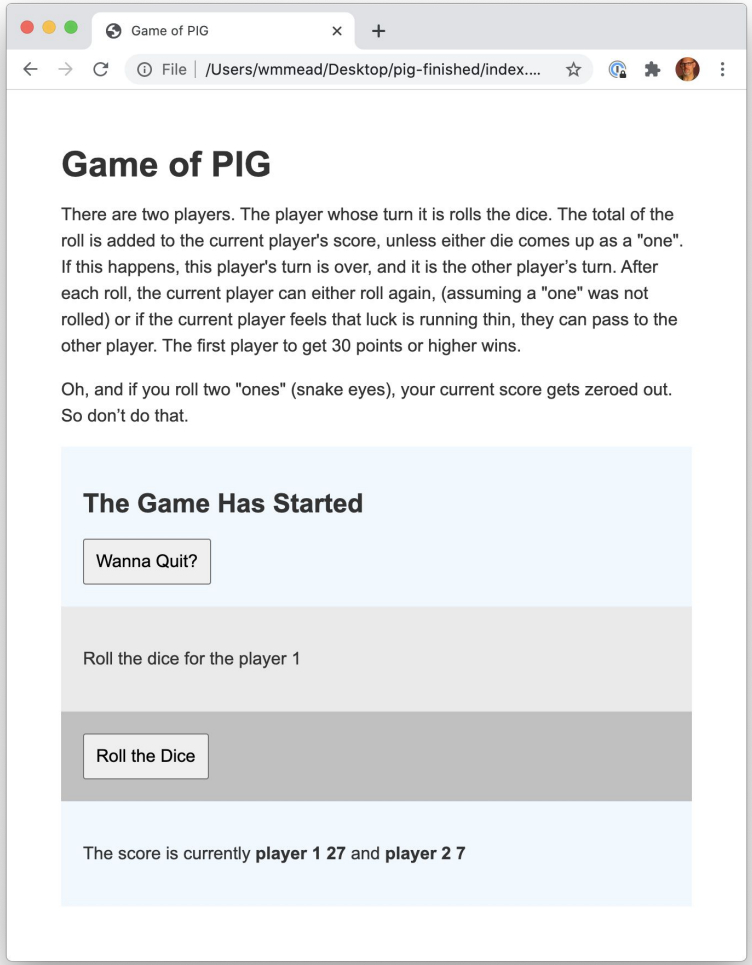
Also, be sure to call the checkWinningCondition() function in the else statement at the bottom of the throwDice() function.

```
function showCurrentScore() {
    score.innerHTML = `<p>The score is currently <strong>${gameData.players[0]}
    ${gameData.score[0]}</strong> and <strong>${gameData.players[1]}
    ${gameData.score[1]}</strong></p>`;
}
```

Final Clean Up

Congratulations, you now have a working, but not very interesting game. As a final step:

1. move the JavaScript to a separate linked file,
2. put it inside a IIFE closure,
3. add the 'use strict' directive
4. and change the variable declarations from var to const.



Extending the Game

Easy Ways to Extend the Game

1. When you roll a one, it zeros out your cumulative score for that turn.
2. Better graphics / interface for the game
3. Create form fields for the player's names and store those in the object.
4. Replace the dice with cards, and potentially have more than six values.
5. Allow player to set the threshold for a win at the beginning of the game.

More Challenging Ways to Extend the Game

1. Add more players.
2. Add more dice/card on each turn/roll.
3. Add more conditions that affect the gameplay/score.

Can you come up with other ideas?