var aQuestion = {

question: "What is 2 \* 5?",

A: "12",

B: "9",

C: "15",

D: "10",

correctAnswer: "D",

explainer: "2 x 5 = 10",

};

var bQuestion = {

question: "What is 7 - 8?",

A: "-2",

B: "1",

C: "-1",

D: "0",

correctAnswer: "C",

explainer: "7- 8 = -1",

};

var cQuestion = {

question: "What is 16/8?",

A: "-2",

B: "2",

C: "-1",

D: "0",

correctAnswer: "B",

explainer: "16/8 = 2",

};

var dQuestion = {

question: "Click button A",

A: "click here",

B: "not here",

C: "nor here",

D: "not this......<br>....",

correctAnswer: "A",

explainer: "It said click button A",

};

var eQuestion = {

question: "What is 16 % 5?",

A: "-2",

B: "1",

C: "-1",

D: "0",

correctAnswer: "B",

explainer: "The remainder of 5 into 16 is 1",

};

var sampleQuestions = [aQuestion, bQuestion, cQuestion, dQuestion, eQuestion]

function makeQuestion(newQuestion, ans1, ans2, ans3, ans4, correctAns){

this.question = newQuestion;

this.A = ans1;

this.B = ans2;

this.C = ans3;

this.D = ans4;

this.correctAnswer = correctAns;

}

var myGlobalTime;

var questionTime = 10;

var showAnswerTime = 3;

function playGame(sampleQuestionArr, aPlayer) {

var i = -1;

console.log("playGame");

disableAnsButtons();

disableQuit();

$("#timer-display").hide();

$("#question").hide();

$("#play-btn").on("click", function() {

playBtnClick(sampleQuestionArr, i, aPlayer);

});

}

function disableAnsButtons() {

$("#answer-1-btn").prop("disabled", true);

$("#answer-2-btn").prop("disabled", true);

$("#answer-3-btn").prop("disabled", true);

$("#answer-4-btn").prop("disabled", true);

}

function enableAnsButtons() {

$("#answer-1-btn").prop("disabled", false);

$("#answer-2-btn").prop("disabled", false);

$("#answer-3-btn").prop("disabled", false);

$("#answer-4-btn").prop("disabled", false);

}

function disableQuit() {

$("#quit-btn").prop("disabled", true);

}

function enableQuit() {

$("#quit-btn").prop("disabled", false);

}

function disableAllButtons() {

disableAnsButtons();

$("#play-btn").prop("disabled", true);

$("#quit-btn").prop("disabled", true);

}

function enableAllButtons() {

enableAnsButtons();

$("#play-btn").prop("disabled", false);

$("#quit-btn").prop("disabled", false);

}

function isAnswerCorrect(questionArray\_iAC, index\_iAC, whichButtonClicked) {

console.log("isAnswerCorrect called")

if (whichButtonClicked == questionArray\_iAC[index\_iAC].correctAnswer) {

return true;

} else {

return false;

}

}

function displayQuestion(question\_dQ) {

$("#question").html(question\_dQ.question);

$("#answerA").html(question\_dQ.A);

$("#answerB").html(question\_dQ.B);

$("#answerC").html(question\_dQ.C);

$("#answerD").html(question\_dQ.D);

$("#correctAnswer").html(question\_dQ.correctAns);

}

function nextQExist(questionsArray\_nQE, index\_nQE) {

if (index\_nQE < questionsArray\_nQE.length-1) {

return true;

} else {

return false;

}

}

function lastQuestion(questions\_lQ, index\_lQ) {

if (index\_lQ < questions\_lQ.length-1) {

return true;

} else {

return false;

}

}

function quit() {

console.log("quit() called");

if (confirm("Do you really want to quit?")) {

doneGame();

} else {

//return to game

}

}

function doneGame() {

console.log("doneGame called\n\tabout to call stop clock");

// stopClock();

disableAnsButtons();

disableQuit();

$("#timer-display").hide();

// $("#instructions-to-player").show();

$("#play-lbl").text("Play Again?");

$("#theAnswer").html("<h1>Game Over!</h1>");

$("#theAnswer").append("<h1>Good Bye!</h1>");

}

function updatePlayer(question\_upd, curLetter, wasRight, thePlayer) {

thePlayer.playerAnswers[thePlayer.playerAnswers.length] = curLetter;

if (wasRight) {

thePlayer.playerScore++;

thePlayer.playerRight++;

thePlayer.rightAnswers[thePlayer.playerAnswers.length] = curLetter;

} else {

var answerShouldBe = question\_upd.correctAnswer;

thePlayer.playerScore--;

thePlayer.playerWrong++;

thePlayer.rightAnswers[thePlayer.playerAnswers.length] = answerShouldBe;

}

}

function displayPlayerStats(thePlayer) {

}

function evalAnswer(questionsArray\_eA, index\_eA, letterChosen, somePlayer) {

console.log("evalAnswer called");

// stopClock();

var answerShouldBe = questionsArray\_eA[index\_eA].explainer;

var answerString = "something wrong in answer string!";

if (isAnswerCorrect(questionsArray\_eA, index\_eA, letterChosen)) {

answerString = "You chose "+letterChosen+". You're right! ";

updatePlayer(questionsArray\_eA[index\_eA], letterChosen, true, somePlayer);

} else {

// youreWrong(questionsArray\_eA, index\_eA, letterChosen, somePlayer);

if (letterChosen != ' '){

answerString = "You chose "+letterChosen+". You're Wrong, Dude! ";

} else {

answerString = "You didn't answer. That's Wrong, Dude! ";

}

updatePlayer(questionsArray\_eA[index\_eA], letterChosen, false, somePlayer);

}

answerString += "<br><span style='color: red'>The right answer is "+answerShouldBe+"<span>";

$("#timer-dislplay").hide();

$("#the-answer").show();

$("#the-answer").html(answerString);

}

function outOfTime(questionsArray\_oOT, index\_oOT, inPlayer) {

console.log("outOfTime called");

evalAnswer(questionsArray\_oOT, index\_oOT, ' ', inPlayer);

}

function resetAfterAnswer(questionsArray\_rAA, index\_rAA, player) {

// stopClock();

$("#the-answer").hide();

$("#timer-disply").show();

console.log("resetAfterAnswer called\n\tabout to call stop clock");

if (nextQExist(questionsArray\_rAA, index\_rAA)) {

// $("#theAnswer").html("Answer Will Display Here");

enableAnsButtons();

askQuestions(questionsArray\_rAA, index\_rAA+1, player);

} else {

doneGame();

}

}

function askQuestions(questionArray\_aQ, index\_aQ, player) {

var theCount = questionTime+showAnswerTime;

console.log("\*\*\*\*\*\*\*\*\*\naskQuestion called\n\tQ No. "+index\_aQ+" asked\n\*\*\*\*\*\*\*");

//universal time loop

$("#timer").show();

$("#the-answer").hide()

myGlobalTime = setInterval( function() {

console.log("the count is "+theCount)

if (theCount > showAnswerTime) {

//keep showing question

//update timer

$("#timer").text(theCount);

theCount--;

} else if (theCount > 0 && theCount <= showAnswerTime) {

$("#timer-display").hide();

$("#the-answer").show();

// clearInterval(myGlobalTime);

evalAnswer(questionArray\_aQ, index\_aQ, ' ', player);

} else if (theCount == 0) {

resetAfterAnswer(questionArray\_aQ, index\_aQ, player);

} else {

console.log("this sucks");

}

}, 1000);

displayQuestion(questionArray\_aQ[index\_aQ]);

$("#answer-1-btn").on("click", function() {

console.log("A1 about to call stop clock");

$("#timer-display").hide();

disableAnsButtons();

evalAnswer(questionArray\_aQ, index\_aQ, "A", player);

// resetAfterAnswer(questionArray\_aQ, index\_aQ, player);

});

$("#answer-2-btn").on("click", function() {

console.log("A2 about to call stop clock");

$("#timer-display").hide();

disableAnsButtons();

evalAnswer(questionArray\_aQ, index\_aQ, "B", player);

// resetAfterAnswer(questionArray\_aQ, index\_aQ, player);

})

$("#answer-3-btn").on("click", function() {

console.log("A3 about to call stop clock");

$("#timer-display").hide();

disableAnsButtons();

evalAnswer(questionArray\_aQ, index\_aQ, "C", player);

// resetAfterAnswer(questionArray\_aQ, index\_aQ, player);

});

$("#answer-4-btn").on("click", function() {

console.log("A4 about to call stop clock");

$("#timer-display").hide();

disableAnsButtons();

evalAnswer(questionArray\_aQ, index\_aQ, "D", player);

// resetAfterAnswer(questionArray\_aQ, index\_aQ, player);

});

$("#quit-btn").on("click", function () {

console.log("quit button about to call stop clock");

quit();

});

$("#play-btn").on("click", function() {

playBtnClick(questionArray\_aQ, index\_aQ, player);

});

}

function playBtnClick(questionArray\_pBC, index\_pBC, aPlayer) {

console.log("playBtnClick");

var btnStatus = $("#play-lbl").html();

switch(btnStatus) {

case "Play":

// console.log("you clicked Play!");

$("#instructions-to-player").hide();

$("#timer-display").show();

enableAnsButtons();

enableQuit();

$("#question").show();

$("#play-lbl").text("Next");

askQuestions(questionArray\_pBC, index\_pBC+1, aPlayer);

break;

case "Next":

// console.log("you clicked Next!");

evalAnswer(questionArray\_pBC, index\_pBC, " ", aPlayer);

resetAfterAnswer(questionArray\_pBC, index\_pBC, aPlayer);

// doneGame();

break;

case "Play Again?":

location.reload();

// $("body").html("<h1>Not Yet Implemented</h1>")

break;

default:

console.log("something's not working in playBtnClick")

}

}