**blackjack**

**grid code**

// ======= ======= ======= updateGameState ======= ======= =======

Sequencer.prototype.updateGameState = function(whichState) {

console.log("");

console.log("== updateGameState ==");

...

// == remove prev items; add next items

for (var j = 0; j < removeItemsArray.length; j++) {

nextItem = removeItemsArray[j];

display.modifyGridRegion(nextItem, "prev");

}

for (var j = 0; j < addItemsArray.length; j++) {

nextItem = addItemsArray[j];

display.modifyGridRegion(nextItem, "next");

}

...

// ======= ======= ======= loadNewPlayer ======= ======= =======

Game.prototype.loadNewPlayer = function() {

console.log("loadNewPlayer");

var playerCount = game.playerNamesArray.length;

var currentPlayer = game.playerObjectsArray[playerCount - 1];

var playerParamsArray = [currentPlayer.bgParams.borderH, currentPlayer.bgParams.borderV, currentPlayer.btnParams.betOnesBtn, currentPlayer.btnParams.betFivesBtn, currentPlayer.btnParams.betTensBtn, currentPlayer.textParams.pName, currentPlayer.textParams.pScore, currentPlayer.textParams.pBank, currentPlayer.textParams.pBet\_1s, currentPlayer.textParams.pBet\_5s, currentPlayer.textParams.pBet\_10s];

for (var j = 0; j < playerParamsArray.length; j++) {

nextItem = playerParamsArray[j];

if (nextItem != null) {

display.modifyGridRegion(nextItem, "next");

// == player buttons not active yet (display amounts only)

if (nextItem.type == "btn") {

var indexCell = display.tableCellsArray[nextItem.iR][nextItem.iC];

sequencer.deActivateButton(indexCell, "click")

}

}

}

currentPlayer.textParams.pName.value = this.name;

currentPlayer.textParams.pScore.value = 0;

currentPlayer.textParams.pBank.value = currentPlayer.totalBank;

}

// ======= ======= ======= modifyGridRegion ======= ======= =======

Display.prototype.modifyGridRegion = function(whichItem, prevNext) {

console.log("modifyGridRegion");

var whichMerge, buttonActivate;

if (prevNext == "prev") {

if (whichItem.merge == "merge") {

whichMerge = "unmerge";

} else {

whichMerge = "restore";

}

buttonActivate = false;

} else {

if (whichItem.merge == "merge") {

whichMerge = "merge";

} else {

whichMerge = null;

}

buttonActivate = true;

}

var whichType = whichItem.type;

var whichValue = whichItem.value;

var whichName = whichItem.name;

console.log(" whichItem.name: " + whichItem.name);

if (whichMerge == "merge") {

indexCell = this.mergeRegion(whichItem);

} else if (whichMerge == "unmerge") {

indexCell = this.unMergeRegion(whichItem);

} else if (whichMerge == "restore") {

indexCell = this.deselectTableCells(whichItem);

} else {

indexCell = this.selectTableCells(whichItem);

}

// ======= data connection =======

switch(whichType) {

case "bg":

break;

case "btn":

if (buttonActivate == true) {

$(indexCell).text(whichValue);

sequencer.activateButton(indexCell, whichItem.callback)

console.log(" whichItem.callback: " + whichItem.callback);

} else {

sequencer.deActivateButton(indexCell, whichItem.callback)

}

break;

case "text":

$(indexCell).text(whichValue);

break;

case "input":

newTextInput = "<input id='" + whichItem.name + "Input' class='" + whichItem.class + "' type='text' value='Tom'>"

$(indexCell).append(newTextInput);

$(newTextInput).attr("id", whichItem.name);

case "image":

break;

}

}

// ======= ======= ======= mergeRegion ======= ======= =======

Display.prototype.mergeRegion = function(whichItem, offsetR, offsetC) {

console.log("mergeRegion");

if (!offsetR) { offsetR = 0 };

if (!offsetC) { offsetC = 0 };

var offsetR = parseInt(offsetR);

var offsetC = parseInt(offsetC);

var indexCell = this.tableCellsArray[whichItem.iR + offsetR][whichItem.iC + offsetC];

for (var j = 0; j < whichItem.iH; j++) {

for (var i = 0; i < whichItem.iW; i++) {

nextCell = this.tableCellsArray[whichItem.iR + offsetR + j][whichItem.iC + offsetC + i];

if (!((i == 0) && (j == 0))) {

$(nextCell).remove();

}

}

}

$(indexCell).attr("colSpan", whichItem.iW);

$(indexCell).attr("rowSpan", whichItem.iH);

$(indexCell).addClass(whichItem.class);

if (whichItem.type != "input") {

$(indexCell).attr("id", whichItem.name);

}

return indexCell;

}

// ======= ======= ======= unMergeRegion ======= ======= =======

Display.prototype.unMergeRegion = function(whichItem, offsetR, offsetC) {

// console.log("unMergeRegion");

if (!offsetR) { offsetR = 0 };

if (!offsetC) { offsetC = 0 };

var offsetR = parseInt(offsetR);

var offsetC = parseInt(offsetC);

var indexCell = this.tableCellsArray[whichItem.iR + offsetR][whichItem.iC + offsetC];

$(indexCell).attr("colSpan", 1);

$(indexCell).attr("rowSpan", 1);

$(indexCell).text("");

$(indexCell).removeClass(whichItem.class);

for (var j = 0; j < whichItem.iH; j++) {

for (var i = 0; i < whichItem.iW; i++) {

if (!((i == 0) && (j == 0))) {

var newCell = document.createElement("td");

$(indexCell).after(newCell);

$(newCell).addClass("cell");

}

}

}

$(indexCell).attr("id", "");

if ($(indexCell).children().length > 0) {

var newCell = document.createElement("td");

$(indexCell).after(newCell);

$(newCell).addClass("cell");

$(indexCell).remove();

}

// return newCell;

}

// ======= ======= ======= selectTableCells ======= ======= =======

Display.prototype.selectTableCells = function(whichItem) {

// console.log("selectTableCells");

var indexCell = this.tableCellsArray[whichItem.iR][whichItem.iC];

for (var j = 0; j < whichItem.iH; j++) {

for (var i = 0; i < whichItem.iW; i++) {

nextCell = this.tableCellsArray[whichItem.iR + j][whichItem.iC + i];

$(nextCell).attr("colSpan", 1);

$(nextCell).attr("rowSpan", 1);

$(nextCell).addClass(whichItem.class);

}

}

$(indexCell).attr("id", whichItem.name);

$(indexCell).addClass(whichItem.class);

return indexCell;

}

// ======= ======= ======= deselectTableCells ======= ======= =======

Display.prototype.deselectTableCells = function(whichItem) {

// console.log("deselectTableCells");

var indexCell = this.tableCellsArray[whichItem.iR][whichItem.iC];

for (var j = 0; j < whichItem.iH; j++) {

for (var i = 0; i < whichItem.iW; i++) {

nextCell = this.tableCellsArray[whichItem.iR + j][whichItem.iC + i];

$(nextCell).attr("colSpan", 1);

$(nextCell).attr("rowSpan", 1);

}

}

$(indexCell).attr("id", "");

$(indexCell).removeClass(whichItem.class);

$(indexCell).empty();

return indexCell;

}