using UnityEngine;

using System.Collections;

using System.Collections.Generic;

public class JSFBombPiece : JSFPieceDefinition {

// only used these functions...

// view more ovveridable functions in PieceDefinition.cs script itself OR

// check out aPieceTemplate.cs script

public override bool powerMerge (List<JSFBoard> chain, JSFGamePiece target,

JSFGamePiece refPiece, int[] arrayRef, bool isACheck)

{

if(target.pd is JSFVerticalPiece || target.pd is JSFHorizontalPiece){

if(!isACheck) doPowerMergeX(arrayRef,1,target,refPiece); // do a power merge power

return true; // <--- has power merge

}

if(target.pd is JSFBombPiece){

if(!isACheck) StartCoroutine( doPowerMergeXX(arrayRef,2, target, refPiece) ); // do a power merge power

return true; // <--- has power merge

}

return false; // <--- no power merge

}

public override bool createPowerAtSwipeEnd (JSFGamePiece gp, int swipeLength)

{

if(swipeLength >= gm.minSwipeMatch + 3){ // meet min swipe + 3 length

gp.master.convertToSpecial(this); // convert to this power :)

gp.master.panelHit(); // hits the panel as well...

return true;

}

return false;

}

public override bool performPower (int[] arrayRef)

{

doBombPower(arrayRef,1); // do bomb power with radius of 1

// return calls...

return false; // default call - will destroy the piece immediately after this ( AKA after calling the power )

// return true; // Only if you do not wish the custom piece to be destroyed instantly

/\*

\* IMPORTANT :

\* you must call gm.destroyInTimeMarked(arrayRef, delay, mScore);

\* either here or in your power function to manually destroy the piece

\* ( that is if you returned false; )

\*

\*/

}

public void doBombPower(int[] arrayRef, int radius){

gm.animScript.doAnim(JSFanimType.BOMB,arrayRef[0],arrayRef[1]); // perform anim

gm.audioScript.bombSoundFx.play(); // play arrow sound fx

// all the surrounding neighbour boards...

foreach(JSFBoard \_board in gm.getBoardsFromDistance(arrayRef,radius) ){

gm.destroyInTime(\_board.arrayRef,0.1f,scorePerPiece);

}

}

// power merge ability code

IEnumerator doPowerMergeXX(int[] arrayRef,int radius, JSFGamePiece target, JSFGamePiece refPiece){

gm.destroyInTimeMarked(target.master,2.1f,scorePerPiece);

gm.destroyInTimeMarked(refPiece.master,2.1f,scorePerPiece);

// visual effect for a time bomb

Vector3 newSize = Vector3.Scale(refPiece.thisPiece.transform.localScale,new Vector3(1.45f,1.45f,1f));

LeanTween.scale( target.thisPiece, newSize ,0.5f).setLoopPingPong();

LeanTween.scale( refPiece.thisPiece, newSize ,0.5f).setLoopPingPong();

target.thisPiece.GetComponent<JSFPieceTracker>().enabled = false;

refPiece.thisPiece.GetComponent<JSFPieceTracker>().enabled = false;

doPowerMergeX(target.master.arrayRef,radius,target,refPiece); // blast with arrows flying!

doBombPower(refPiece.master.arrayRef,radius); // normal big blast w/0 arrows

yield return new WaitForSeconds(2f);

doBombPower(target.master.arrayRef,1); // normal blast

doBombPower(refPiece.master.arrayRef,1); // normal blast

}

// power merge ability code

void doPowerMergeX(int[] arrayRef,int radius, JSFGamePiece target, JSFGamePiece refPiece){

gm.destroyInTimeMarked(target.master,0f,scorePerPiece);

gm.destroyInTimeMarked(refPiece.master,0f,scorePerPiece);

doBombPower(arrayRef,radius); // do bomb power with specified radius

// arrow power...

float delay = 0f; // the delay variable we are using...

float delayIncreament = 0.1f; // the delay of each piece being destroyed.

gm.audioScript.arrowSoundFx.play(); // play arrow sound fx

bool destroyThis = false; // variable to help skip the first board in the list

// the top of this board...

foreach(JSFBoard \_board in gm.iBoard(arrayRef).getAllBoardInDirection(JSFBoardDirection.Top) ){

if(destroyThis) gm.destroyInTime(\_board.arrayRef,delay,scorePerPiece);

destroyThis = true;

delay += delayIncreament;

}

delay = 0f; // reset the delay

destroyThis = false; // help to skip the first board in the list...

// the bottom of this board...

foreach(JSFBoard \_board in gm.iBoard(arrayRef).getAllBoardInDirection(JSFBoardDirection.Bottom) ){

if(destroyThis) gm.destroyInTime(\_board.arrayRef,delay,scorePerPiece);

destroyThis = true;

delay += delayIncreament;

}

switch(gm.boardType){

case JSFBoardType.Hexagon :

gm.animScript.doAnim(JSFanimType.ARROWTLBR,arrayRef[0],arrayRef[1]); // perform anim

gm.animScript.doAnim(JSFanimType.ARROWTRBL,arrayRef[0],arrayRef[1]); // perform anim

delay = 0f; // reset the delay

destroyThis = false; // help to skip the first board in the list...

// the TopLeft of this board...

foreach(JSFBoard \_board in gm.iBoard(arrayRef).getAllBoardInDirection(JSFBoardDirection.TopLeft) ){

if(destroyThis) gm.destroyInTime(\_board.arrayRef,delay,scorePerPiece);

destroyThis = true;

delay += delayIncreament;

}

delay = 0f; // reset the delay

destroyThis = false; // help to skip the first board in the list...

// the TopRight of this board...

foreach(JSFBoard \_board in gm.iBoard(arrayRef).getAllBoardInDirection(JSFBoardDirection.TopRight) ){

if(destroyThis) gm.destroyInTime(\_board.arrayRef,delay,scorePerPiece);

destroyThis = true;

delay += delayIncreament;

}

delay = 0f; // reset the delay

destroyThis = false; // help to skip the first board in the list...

// the BottomLeft of this board...

foreach(JSFBoard \_board in gm.iBoard(arrayRef).getAllBoardInDirection(JSFBoardDirection.BottomLeft) ){

if(destroyThis) gm.destroyInTime(\_board.arrayRef,delay,scorePerPiece);

destroyThis = true;

delay += delayIncreament;

}

delay = 0f; // reset the delay

destroyThis = false; // help to skip the first board in the list...

// the BottomRight of this board...

foreach(JSFBoard \_board in gm.iBoard(arrayRef).getAllBoardInDirection(JSFBoardDirection.BottomRight) ){

if(destroyThis) gm.destroyInTime(\_board.arrayRef,delay,scorePerPiece);

destroyThis = true;

delay += delayIncreament;

}

break;

case JSFBoardType.Square :

gm.animScript.doAnim(JSFanimType.ARROWVH,arrayRef[0],arrayRef[1]); // perform anim

delay = 0f; // reset the delay

destroyThis = false; // help to skip the first board in the list...

// the Left of this board...

foreach(JSFBoard \_board in gm.iBoard(arrayRef).getAllBoardInDirection(JSFBoardDirection.Left) ){

if(destroyThis) gm.destroyInTime(\_board.arrayRef,delay,scorePerPiece);

destroyThis = true;

delay += delayIncreament;

}

delay = 0f; // reset the delay

destroyThis = false; // help to skip the first board in the list...

// the Right of this board...

foreach(JSFBoard \_board in gm.iBoard(arrayRef).getAllBoardInDirection(JSFBoardDirection.Right) ){

if(destroyThis) gm.destroyInTime(\_board.arrayRef,delay,scorePerPiece);

destroyThis = true;

delay += delayIncreament;

}

if(gm.iBoard(arrayRef).top != null){

gm.animScript.doAnim(JSFanimType.ARROWH,gm.iBoard(arrayRef).top.arrayRef); // perform anim

delay = 0f; // reset the delay

destroyThis = false; // help to skip the first board in the list...

// up+1 & left

foreach(JSFBoard \_board in gm.iBoard(arrayRef).top.getAllBoardInDirection(JSFBoardDirection.Left) ){

if(destroyThis) gm.destroyInTime(\_board.arrayRef,delay,scorePerPiece);

destroyThis = true;

delay += delayIncreament;

}

delay = 0f; // reset the delay

destroyThis = false; // help to skip the first board in the list...

// up+1 & right

foreach(JSFBoard \_board in gm.iBoard(arrayRef).top.getAllBoardInDirection(JSFBoardDirection.Right) ){

if(destroyThis) gm.destroyInTime(\_board.arrayRef,delay,scorePerPiece);

destroyThis = true;

delay += delayIncreament;

}

}

if(gm.iBoard(arrayRef).bottom != null){

gm.animScript.doAnim(JSFanimType.ARROWH,gm.iBoard(arrayRef).bottom.arrayRef); // perform anim

delay = 0f; // reset the delay

destroyThis = false; // help to skip the first board in the list...

// down+1 & left

foreach(JSFBoard \_board in gm.iBoard(arrayRef).bottom.getAllBoardInDirection(JSFBoardDirection.Left) ){

if(destroyThis) gm.destroyInTime(\_board.arrayRef,delay,scorePerPiece);

destroyThis = true;

delay += delayIncreament;

}

delay = 0f; // reset the delay

destroyThis = false; // help to skip the first board in the list...

// down+1 & right

foreach(JSFBoard \_board in gm.iBoard(arrayRef).bottom.getAllBoardInDirection(JSFBoardDirection.Right) ){

if(destroyThis) gm.destroyInTime(\_board.arrayRef,delay,scorePerPiece);

destroyThis = true;

delay += delayIncreament;

}

}

if(gm.iBoard(arrayRef).left != null){

gm.animScript.doAnim(JSFanimType.ARROWV,gm.iBoard(arrayRef).left.arrayRef); // perform anim

delay = 0f; // reset the delay

destroyThis = false; // help to skip the first board in the list...

// left+1 & up

foreach(JSFBoard \_board in gm.iBoard(arrayRef).left.getAllBoardInDirection(JSFBoardDirection.Top) ){

if(destroyThis) gm.destroyInTime(\_board.arrayRef,delay,scorePerPiece);

destroyThis = true;

delay += delayIncreament;

}

delay = 0f; // reset the delay

destroyThis = false; // help to skip the first board in the list...

// left+1 & down

foreach(JSFBoard \_board in gm.iBoard(arrayRef).left.getAllBoardInDirection(JSFBoardDirection.Bottom) ){

if(destroyThis) gm.destroyInTime(\_board.arrayRef,delay,scorePerPiece);

destroyThis = true;

delay += delayIncreament;

}

}

if(gm.iBoard(arrayRef).right != null){

gm.animScript.doAnim(JSFanimType.ARROWV,gm.iBoard(arrayRef).right.arrayRef); // perform anim

delay = 0f; // reset the delay

destroyThis = false; // help to skip the first board in the list...

// right+1 & up

foreach(JSFBoard \_board in gm.iBoard(arrayRef).right.getAllBoardInDirection(JSFBoardDirection.Top) ){

if(destroyThis) gm.destroyInTime(\_board.arrayRef,delay,scorePerPiece);

destroyThis = true;

delay += delayIncreament;

}

delay = 0f; // reset the delay

destroyThis = false; // help to skip the first board in the list...

// right+1 & down

foreach(JSFBoard \_board in gm.iBoard(arrayRef).right.getAllBoardInDirection(JSFBoardDirection.Bottom) ){

if(destroyThis) gm.destroyInTime(\_board.arrayRef,delay,scorePerPiece);

destroyThis = true;

delay += delayIncreament;

}

}

break;

}

}

}