**In JSFGameManager.cs**

// suggest a piece after a given time...

IEnumerator suggestPiece() {

yield return new WaitForSeconds(suggestionTimer); // wait till it's time

if(gameState != GameState.GameActive){

yield break; // game no longer active... do not display suggestion...

}

foreach(GameObject go in suggestedPieces){

if(go == null || !go.activeSelf){

notifyBoardHasChanged(); // something changed... perform checks again!

yield break;

}

float currentSize = pieceOriginalSize.x;

// main scaler loop

LeanTween.value(go,suggestPieceScaler,currentSize\*0.75f,currentSize\*1.25f,1f)

.setLoopPingPong().setOnUpdateParam(go);

// sub rotate loop

go.transform.localEulerAngles = new Vector3(0,0,340f);

LeanTween.rotateZ(go,20f,0.8f).setLoopPingPong();

}

}

**In JSFSwipeManager.cs**

// function to validate the swipe chain

public static void validateSwipe(){

bool validateStatus = false;

int swipeNum = currentSwipeCount;

int comboTracker = 0;

if( gm.gameState == JSFGameState.GameActive && // game has not ended...

(swipeChain.Count >= gm.minSwipeMatch || hasPowerMerge)){ // meet the minimum required length

gm.audioScript.matchSoundFx.play();

gm.playerMadeAMove();

if(hasPowerMerge){ // perform powerMerges if there is...

List<JSFBoard> mergeCheck = new List<JSFBoard>(); // a power merge check list

bool powerMerged = false;

for(int w = swipeChain.Count-1; w >= 0; w-- ){

mergeCheck.Clear(); // clear the list for each check

mergeCheck.AddRange(swipeChain); // copy the current list

mergeCheck.Remove(swipeChain[w]); // as to not check itself

mergeCheck.Reverse(); // reverse the list to start from last swiped

foreach(JSFBoard innerLoop in mergeCheck){ // check against all swiped pieceDefinitions

if(innerLoop.pd.powerMerge(swipeChain,swipeChain[w].piece,innerLoop.piece,swipeChain[w].arrayRef,false) ||

swipeChain[w].pd.powerMerge(swipeChain,innerLoop.piece,swipeChain[w].piece,swipeChain[w].arrayRef,false) ){

powerMerged = true; // powerMerge called...

break; // done power merging... ( break inner loop )

}

}

if(powerMerged){

break; // done power merging... ( break outer loop too )

}

}

}

foreach(JSFBoard board in swipeChain){

comboTracker++;

if(comboTracker > gm.minSwipeMatch){

gm.increaseCombo(); // increases combo by 1 for each after the min swipe num

}

if(board.isFilled){ // if not already destroyed by power merge

JSFRelay.onSwipeValidating(board.piece); // relay call

}

}

validateStatus = true; // swipe chain has been validated...

} else if(swipeNum > 1) {

gm.audioScript.badMoveSoundFx.play();

}

swipeChain.Clear();

removeSwipeVisuals(0);

limitedSwipe = false; // reset swipe limit

hasPowerMerge = false; // reset powerMerge status

rainbowSwipe = false; // resets the any swipe status

isSwiping = false; // disable swiping after validated

JSFRelay.onSwipeValidated(validateStatus,swipeNum); // relay call

}

**\*\*New Script\*\* JSFGlobalDefines**

**Put in “Assets\JSF\Scripts\Area 51\Editor\JSF Related”**

using UnityEditor;

using UnityEngine;

using System.Collections;

[InitializeOnLoad]

public static class JSFGlobalDefines {

static JSFGlobalDefines()

{

string defines;

foreach(BuildTargetGroup btg in System.Enum.GetValues(typeof(BuildTargetGroup))){

defines = PlayerSettings.GetScriptingDefineSymbolsForGroup(btg);

if(!defines.Contains("JSF")){

// adds the JSF global define to the project :)

PlayerSettings.SetScriptingDefineSymbolsForGroup(btg,defines + ";JSF");

}

}

}

}