Example Skeleton Classes for Rock, Paper, Scissors

**RPSCcontroller:**

```java
/**
 * RPSCcontroller is the controller for the Rock, Paper, Scissors game
 * It takes input from the user and changes the model accordingly
 * It also changes the view to reflect the model
 */

class RPSCcontroller implements KeyListener {
    
    // create an instance of the model in the RPSCcontroller
    RPSCLogic model;
    // create an instance of the view in the RPSCView
    RPSCView view;

    /**
     * Constructor. Sets up model and view instance variables.
     */
    public RPSCcontroller() {

    }

    /**
     * Necessary for KeyListener
     * Listens for key input from user.
     * Calls updateModel and updateView
     */
    public void keyTyped(KeyEvent e) {

    }
```
/**
  * Updates model with userInput
  * Calls whoWon and getScore from model.
  */
public void updateModel(int userInput) {
}

/**
  * Update view with changed model.
  * calls updateScore and updateMessage in view.
  * @param score current score,
  * @param whoWon true if user won last game, else false
  */
public void updateView(int[] score, boolean whoWon) {
}

/** Necessary for KeyListener implementation */
public void keyPressed(KeyEvent e) {
}

/** Necessary for KeyListener implementation */
public void keyReleased(KeyEvent e) {
}

/**
  * Set-up the model, view, KeyListener
  */
public void main() {
}
RPSLogic:

/**
 * RPSLogic is the model for the Rock, Paper, Scissors game
 * It stores information about the current state of the game.
 * It has functions that can be called by the controller.
 */

public class RPSLogic {

    // keeps the score of user and computer
    private int[] score;

    /**
     * Set-up fields
     */
    public RPSLogic(){
    }

    /**
     * The getScore method returns the current score
     * @return the current score
     */
    public int[] getScore(){
        return null;
    }

    /**
     * whoWon compares the user's choice with the computer's choice
     * and determines the winner. Calls cChoice and updateScore
     * @param uChoice an int corresponding to Rock, Paper, or Scissors
     * @return a boolean that is true if user wins, false if user
     * loses.
     */
    public boolean whoWon(int uChoice){
        return false;
    }

    /**
     * cChoice uses a random number generator to determine the
     */
* computer's choice.
  * @return an int corresponding to rock, paper, or scissors.
  */
private int cChoice() {
  return 0;
}

/**
 * updateScore updates the score in RPSLogic according to who wins
 * @param winner a boolean that is true if the user won, false if
 *                user lost.
 * @return the score.
 */
private int updateScore(boolean winner){
  return 0;
}
RPSView:

/**
 * RPSView is the view for the Rock, Paper, Scissors game
 * It displays information to the user
 * It is updated by the controller
 */

public class RPSView {
    // holds instructions for RPS
    String instructions;
    // holds current message
    String message;

    /**
     * Constructor. Initializes instance variables.
     */
    public RPSView() {
    }

    /**
     * Create the display for the instructions
     */
    public void createDisplay() {
    }

    /**
     * Called by the controller
     * Updates the score seen by user
     */
    public void updateScore(int[] score) {
    }

    /**
     * Called by the controller
     * Updates the message seen by user based on who won last game
     */
    public void updateMessage(boolean whoWon) {
    }
}