```
////////////////////////////////////       class_11_Soloist1_music         //////////////
////  this tab manage the function that govern the user about the first rhythm music

/// import the library that allow to play mp3 files
import ddf.minim.*;

/// this class manage the soloist1 music part of the software:
/// _plays mp3 files
/// _manage the passage between different music channel for each user
/// _set the correct rhytm for each choice
class Soloist_music{
  PApplet parent;
  Minim minim;

  AudioSnippet[] personChannels;

  int numPersonChannels;
  int currentPersonChannel = 0;
  int numChords
  int fadeStepTime = 0;
  float gainUpValue = 0;
  float gainDownValue = 0;
  int fadeUpDelay;

  int[] jump;
  int[] jump_set;
  int[] jump_min;

  int rhytm1;
  int rhytm2;
  int rhytm3;

  int countRhytm1=0;

  Soloist_music() {

    numChords = 7;

    // make the Minim sound object
     minim =new Minim(parent);

    numPersonChannels = 2;
     personChannels =new AudioSnippet[numPersonChannels];

    fadeUpDelay = 500;

  }


  void Play_Soloist_music()  {
      updateChange();
  }
/// this void manage the correct passage between rhytms: set the fade out and the fade in of the sound everytime the user change po
  void updateChange()  {



    if (millis() > fadeStepTime) {

       fadeStepTime =millis() + (fadeUpDelay/46);
      if (currentPersonChannel == 0) {

        if (personChannels[0]  !=null) {

          if (gainUpValue < 6) {

            gainUpValue++;
            personChannels[0].setGain(gainUpValue);
          }
        }
        if (personChannels[1]  !=null) {
          if (gainDownValue > -40) {

            gainDownValue--;
            personChannels[1].setGain(gainDownValue);
          }
         }
       }
```

```
      else if (currentPersonChannel == 1) {
        if (personChannels[1]  !=null) {
          if (gainUpValue < 6) {

            gainUpValue++;
            personChannels[1].setGain(gainUpValue);
          }
        }
        if (personChannels[0]  !=null) {
          if (gainDownValue > -40) {

            gainDownValue--;
            personChannels[0].setGain(gainDownValue);
          }
         }
      }
     }
  }


  void changeRhytm(int rhytmType,int  baseChord,int BaseTone) {


    // choose which rhythm to play

    if (currentPersonChannel == 0) {

      if (personChannels[0]  !=null) {
        gainDownValue = 6;
      }

 //// selects  three  different rhythm for the user's choice
       rhythm1 =int(random(0,2));

       rhythm2 =int(2);

       rhythm3 =int(random(3,5));

/// here the software load the correct mp3 file for each choice (for the channel 1)
      switch(rhytmType) {
      case  1:
        personChannels[1] = Rhythm [baseChord][0][rhythm1][BaseTone];
        break;

      case  2:
         personChannels[1] = Rhythm [baseChord][0][rhythm2][BaseTone];
        break;

      case  3:
        personChannels[1] = Rhythm [baseChord][0][rhythm3][BaseTone];
        break;
      }


      gainUpValue = -40;
      personChannels[1].setGain(gainUpValue);
      personChannels[1].loop();


      currentPersonChannel = 1;
     }
    else if (currentPersonChannel == 1) {

      if (personChannels[1]  !=null) {
        gainDownValue = 6;
      }

       rhytm1 =int(random(0,2));

       rhytm2 =int(2);

       rhytm3 =int(random(3,5));

/// here the software load the correct mp3 file for each choice (for the channel 0)
      switch(rhytmType) {
      case  1:
        personChannels[0] = Rhythm [baseChord][0][rhytm1][BaseTone];
        break;
```

```
      case  2:
        personChannels[0] =Rhythm [baseChord][0][rhytm2][BaseTone];
        break;
      case  3:
        personChannels[0] =Rhythm [baseChord][0][rhytm3][BaseTone];
        break;
      }

      gainUpValue = -40;
      personChannels[0].setGain(gainUpValue);
      personChannels[0].loop();
      currentPersonChannel = 0;
    }
  }

  void PauseSound(){
    if (personChannels[0]  !=null) {
      personChannels[0].pause();
      personChannels[0].rewind();
    }
    if (personChannels[1]  !=null) {
      personChannels[1].pause();
      personChannels[1].rewind();
    }

  }

  void stop(){
    minim.stop();
    PauseSound();
  }

}
```