设置参数说明书

这里给大家说明一下配置文件config.py里的每个参数对应的含义，以便大家修改的时候更加明确。

目录

屏幕相关参数

screen_size: 屏幕尺寸（宽度，高度）

background_image: 背景图片文件路径

background_size: 背景图片尺寸

piano_image: 钢琴图片文件路径

piano_size: 钢琴图片尺寸

message_color: 和弦显示的字体颜色，格式为(R, G, B, A)

fonts_size: 字体大小

label1_place: 显示当前演奏的音符名称的文字位置

label2_place: 显示和弦名称的文字位置

label3_place: 播放midi文件的状态文字位置

label_anchor_x: 文字的横向对齐方式

label_anchor_y: 文字的纵向对齐方式

fonts: 字体名称

bold: 是否加粗

notes_image: 音符点模式下音符点的图片路径

notes_resize_num: 音符点的缩放大小

go_back_image: 返回按钮的图片文件路径

go_back_place: 返回按钮的图片位置

self_play_image: 电脑键盘演奏按钮的图片文件路径

self_play_place: 电脑键盘演奏按钮的图片位置

self_midi_image: midi键盘演奏按钮的图片文件路径

self_midi_place: midi键盘演奏按钮的图片位置

play_midi_image: 播放midi文件按钮的图片文件路径

play_midi_place: 播放midi文件按钮的图片位置

key_settings: 电脑键盘对应88键的字典，请注意音源路径里的文件必须要全部包含你所设置到的键位

midi_device_id: 这个参数是接入midi设备后（比如midi键盘），midi设备所对应的id

pygame mixer初始化参数

frequency, size, channel, buffer, maxinum_channels

键盘演奏以及播放MIDI文件的相关参数

pause_key: 暂停的按键

repeat_key: 重复播放的按键

unpause_key: （暂停中）继续的按键

exit_key: 退出程序的按键

global_volume: 总音量大小，最大为1，最小为0

delay: 是否在弹一个音放开之后给音一定的延迟

delay_time: 延迟的时间（秒）

fadeout_ms: 音符播放结束后渐出所用的时间，单位为毫秒

touch_interval: 当连续演奏同一个音的时候，音的结束和重新播放的时间间隔，单位为秒

pause_key_clear_notes: 当暂停的时候是否清除掉当前演奏的所有音符的显示

delay_only_read_current: 当音在延迟播放的时候（音没有被按着），和弦判断不包括这些还在延迟播放的音，只包括当前正在按着的音

sound_format: 音源的文件格式（文件后缀）

sound_path: 音源的文件路径

show_delay_time: 播放midi文件时音符的延迟时间

config_enable: 在电脑键盘演奏时是否开启功能键

config_key: 功能键的键位设置，功能键和其他按键搭配可以做到各种不同的功能

volume_up: 和功能键搭配的按键，让总音量升高

volume_down: 和功能键搭配的按键，让总音量降低

volume_change_unit: 总音量每次变化的音量

change_delay: 和功能键搭配的按键，改变是否延迟

change_read_current: 和功能键搭配的按键，改变是否只判断当前按着的和弦

change_pause_key_clear_notes: 和功能键搭配的按键，改变是否在暂停时清除当前演奏的音的显示

note_place: 音符点模式下，钢琴从左到右全部按键对应的位置

load_sound: 演奏时是否加载音源以及播放（在和宿主一起使用的时候要设置为False）

show_key: 在电脑键盘演奏时是否显示电脑键盘的按键名称

show_chord: 演奏时是否实时通过乐理逻辑分析判断和弦

show_notes: 演奏时是否实时显示当前演奏的音符

get_off_drums: 如果为True，在播放midi文件模式中，如果选择了合并所有音轨，在读取midi文件之后会去掉鼓的轨道，（如果有的话）以避免演示和弦会被鼓的音符扰乱。

sort_invisible: 如果为True，在显示和弦中不会显示排序的内容（比如 `Fmaj7 sort as [2,3,1,4]` 会变成 `Fmaj7`）

play_as_midi: 播放midi文件时不加载音源，直接在软件内部播放midi文件（以midi自带的音源播放），好处是音符比较多的midi文件加载快很多，而且播放时在同时播放的音符很多，和弦类型很复杂的时候不会出现卡顿。设置为True即可进入这个模式。

pitch_range: 两个表示音高的字符串组成的tuple, 当读取MIDI文件时，筛选出音高在这两个音高之间的音符，以防止音符的音高超过钢琴设定的音高范围。

soft_pedal_volume: 在MIDI键盘演奏模式下，踩住弱音踏板时音符的音量减小的比例，0到1之间的小数

和弦乐理分析参数

这几个是和弦判断的乐理逻辑算法的参数，默认的设置适用程度最广泛，如果我要说明这些都是什么意思的话，那可能需要先理解我的算法，因此这一部分等以后我介绍这个算法的时候再说明

detect_mode = 'chord'

inv_num = False

rootpitch = 5

change_from_first = True

original_first = True

same_note_special = False

whole_detect = True (把这个参数改为 `False` 可以提升实时分析的速度，但是非常复杂的和弦可能无法分析出来)

return_fromchord = False

two_show_interval = True

poly_chord_first = False

root_position_return_first = True

alter_notes_show_degree = True

曲子分离主旋律的算法的参数

melody_tol, chord_tol, get_off_overlap_notes, average_degree_length, melody_degree_tol

音符显示相关参数

note_mode: 选择音符显示模式，目前有音符点和音符条（上升）和音符条（下落，只有播放midi文件模式可用）三种模式可以选择，分别对应的是'dots'和'bars'和'bars drop'

bar_width: 音符条的宽度

bar_height: 音符条的长度

bar_color: 音符条的颜色, RGB tuple

sustain_bar_color: 在MIDI键盘演奏模式下，踩住钢琴踏板时按下的键放开后显示的颜色, RGB tuple

bar_y: 音符条出现的纵坐标

bar_offset_x: 音符条偏离音符点位置的横坐标的像素值

bar_opacity: 音符条的透明度，0到255，从完全透明到完全不透明

opacity_change_by_velocity: 透明度是否随着按键力度的变化而变化，按键力度越轻，音符条越透明，按键力度越重，音符条越不透明

color_mode: 音符条的颜色显示模式，目前有单色显示和随机颜色显示两种模式可以选择，分别对应'normal'和'rainbow'（实际上填不是normal的其他文字也可以）

bar_steps: 音符条每次上升移动的像素值

bar_unit: 在播放midi文件的时候，音符条计算相对长度时为单位的长度

bar_hold_increase: 在按住琴键时（或者按住电脑按键时），音符条每次拉长的像素值

bars_drop_interval: 在音符条（下落）模式中，音符条要花多长的时间从屏幕顶端下落到指定位置，单位为秒

bars_drop_place: 在音符条（下落）模式中，音符条下落到的指定位置（高度）

adjust_ratio: 调整音符条下落到指定位置的准确度的一个参数，一般来说不需要修改

bar_border: 音符条的边框宽度

bar_border_color: 音符条的边框颜色，表示RGB的tuple，`(R, G, B)`

钢琴键盘相关参数

draw_piano_keys: 设置为True进入绘制钢琴模式，（按照参数和钢琴88键的结构绘制出钢琴键盘，取代之前的钢琴图片）在绘制钢琴模式下，midi键盘演奏或者电脑键盘演奏时对应的琴键会亮起，包括在下落音符模式播放midi文件时，音符落在琴键上也会亮起。绘制钢琴是采用直接按照钢琴88键的结构，根据可设定的参数绘制出黑白键，并且每个键可以改变颜色。在88键的绘制下面有一张黑色的背景图片，主要用于显示钢琴键之间的缝隙（填充用）。可以关掉音符模式，（note_mode设置为不是dots, bars, bars drop的值即可）只打开绘制钢琴模式，演奏时对应的钢琴键位会亮起，播放midi文件时也会亮起当下的音符。也可以使用任何一个音符模式同时打开绘制钢琴模式。

white_key_width: 钢琴白键的宽度（横向长度）

white_key_height: 钢琴白键的高度（纵向长度）

white_key_interval: 每两个钢琴白键之间的距离

white_key_y: 钢琴白键的高度位置

white_keys_number: 钢琴白键的数量

white_key_start_x: 钢琴第一个白键的横向位置

white_key_color: 钢琴白键的颜色

black_key_width: 钢琴黑键的宽度（横向长度）

black_key_height: 钢琴黑键的高度（纵向长度）

black_key_y: 钢琴黑键的高度位置

black_key_first_x: 钢琴第一个黑键的横向位置

black_key_start_x: 钢琴第二个黑键的横向位置

black_key_color: 钢琴黑键的颜色

black_keys_set: 除了第1个黑键单独设置，每5个黑键为一组，每个组里每个黑键之间的相对间隔（第1个间隔一般为0，表示第1个黑键从当前组里的相对位置的最左边开始）

black_keys_set_interval: 每两个黑键组之间的间隔

black_keys_set_num: 黑键组的个数

piano_background_image: 钢琴底下的背景图片（填充间隙用）

piano_key_border: 钢琴键盘的边框宽度

piano_key_border_color: 钢琴键盘的边框颜色，表示RGB的tuple，`(R, G, B)`

SoundFont文件相关参数

use_soundfont: 是否使用SoundFont文件来播放MIDI文件

play_use_soundfont: 是否在电脑键盘或者MIDI键盘演奏时使用SoundFont文件

sf2_path: SoundFont文件的路径

bank_num: SoundFont文件的bank编号

preset_num: SoundFont文件的preset编号

sf2_duration: SoundFont文件生成的音符的长度，单位为秒

sf2_decay: SoundFont文件生成的音符的渐出时间，单位为秒

sf2_volume: SoundFont文件生成的音符的音量，单位为MIDI音符力度

use_soundfont: 是否使用SoundFont文件来播放MIDI文件

sf2_path: SoundFont文件的路径

显示作曲分析

Ideal Piano可以读取一个具有特定格式的作曲分析的文本文件，在演示midi文件的模式下按照当前小节数来显示曲子的转调，附属和弦，借用和弦等等作曲手法，可以写一个作曲分析的txt文件，然后实时在当下指定的小节显示对应的作曲分析的内容。默认为musical analysis.txt这个文件。

作曲分析文件的格式为：

小节数1

作曲分析内容1

小节数2

作曲分析内容2

小节数3

作曲分析内容3

...

这里的小节数以第1小节作为开头，小节数为一个数字，可以是整数或者小数。作曲分析内容是你想要在进行到指定的小节数的时候想要显示的内容。软件会解析作曲分析文件的格式，然后寻找第一个到当前小节数的音符的位置，然后在演示曲子的时候当进行到对应的音符的位置的时候显示对应的作曲内容。

目前除了这个格式，还支持显示调性，可以在任意的位置显示当前的调性，如果曲子有转调的时候就可以在转调开始的小节之前写调性的语句。语法为（这里是举例）：

key: 调性1（调性这里可以写任意的你想显示的内容，比如A大调, A major等等）

小节数1

作曲分析内容1

小节数2

作曲分析内容2

小节数3

作曲分析内容3

key: 调性2（可以在曲子转调的时候写新的调性）

小节数n

作曲分析内容n

小节数x

作曲分析内容x

小节数y

作曲分析内容y

...

（表示调性的语句与小节数语句之间必须间隔一行，一个小节数语句必须与对应的作曲分析内容语句互为上下相邻行，合在一起称为小节块，不同的小节块之间必须间隔一行。在一个作曲分析内容语句内可以写多行语句，但是中间不能有完全的空行）

（小节数支持绝对小节位置和相对小节位置两种语法，绝对小节位置就是一个数字，可以是整数或者小数，相对小节位置的语法为+相对小节长度，比如+1表示相对上一个位置的下一个小节的位置，+1/2表示相对上一个位置的下一个二分之一小节的位置，相对小节位置支持整数，小数和分数。）

为了能够快速输入大量的作曲分析语句，尤其是在分析一首和弦走向很复杂的曲子的时候，我自己一般来说都是写当前的最新的4-5个和弦然后分成4-5个小节块，每个小节块演奏到其中一个和弦的时候放一个箭头在那个和弦的前面，下面会按照实际情况加上不同的作曲手法讲解，比如

+1

Emaj9(omit 3) | D#m7 | DM7 | → C#11(omit 3)

IVM9 iii7 bIIIM7 V11 (F#大调)

你可以使用music_analysis_batch_language的编辑器，按照README的批处理语法来生成乐理分析语句。

使用这种我设计的批处理输入乐理分析语句的语法，可以非常简洁快速地输入大量的乐理分析语句，而且本身这个乐理分析语句的语法就是可以直接让非程序员看的很舒服的，同时我也写了解析的算法的代码让Ideal Piano这个软件可以解析并且显示出来，因此大家也可以把这个特殊批处理语句当做是写和弦功能分析的一个小小的编程语言来使用，我觉得还是很不错的:D

相关参数

show_music_analysis: 是否开启显示作曲分析内容

music_analysis_file: 读取的作曲分析文件的文件路径

music_analysis_place: 设置显示作曲内容的位置

key_header: 调性开头内容（这个参数是显示调性的开头的内容，比如"当前调性："）

music_analysis_width: 音乐分析文本标签的宽度

music_analysis_fonts_size: 音乐分析文本的字体大小

use_track_colors: 是否使用不同的颜色在不同的音轨和乐器上

tracks_colors: 不同音轨和乐器的颜色的列表，RGB参数

use_default_tracks_colors: 是否使用设定好的音轨的颜色，或者不同的音轨使用随机生成的颜色