

```

1.   ### 444.praat
2.   # mail@cgrauer.de
3.   # 30.05.2024
4.   # All rights reserved
5.   # (c) copyright 2024 Christian Grauer
6.   ###
7.
8.   # get the command line parameters
9.   form Input
10.  word speaker_id SSB1918
11.  word sample_idx 0011
12.  word pos 0
13.  word len 0
14.  sentence pinyin _
15.  endform
16.  sample_id$ = speaker_id$ + sample_idx$
17.  target_id$ = sample_id$ + "_" + pos$
18.
19.  # add target id to dataset
20.  csv$ = target_id$
21.
22.  @log: target_id$ + " - " + "New Target: " + speaker_id$ + "|" + sample_idx$ +
    "|" + pos$
23.
24.  # init global variables
25.  pathSamples$ = "../Data/SLR93.LOC/audio"
26.  pathTargets$ = "../Data/Targets"
27.  pathData$ = "../Data/Targets"
28.  truncpathTarget$ = pathTargets$ + "/" + target_id$ + "/" + target_id$
29.  ext$ = "wav"
30.  fpSample$ = pathSamples$ + "/" + speaker_id$ + "/" + sample_idx$ + "." + ext$
31.  fpTarget$ = truncpathTarget$ + "." + ext$
32.  fpSyllable1$ = truncpathTarget$ + "_s1" + "." + ext$
33.  fpSyllable2$ = truncpathTarget$ + "_s2" + "." + ext$
34.  fpSyllable3$ = truncpathTarget$ + "_s3" + "." + ext$
35.  fpTargetData$ = truncpathTarget$ + "." + "csv"
36.  fpData$ = pathData$ + "/" + "444.csv"
37.  fpTargetsDone$ = pathData$ + "/" + "targets_done.csv"
38.  fpPhpFlag$ = "./php.stop.flag"
39.
40.  # create the target folder if not existing
41.  createFolder: pathTargets$
42.  createFolder: pathTargets$ + "/" + target_id$
43.
44.  # read sample to be analyzed
45.  @log: target_id$ + " - " + "Sample: " + fpSample$
46.  sample = Read from file: fpSample$
47.
48.  # analyze sound sample and store retrieved values
49.  @analyzeSyllable: sample, "sample"
50.
51.  # WAV im Editor anzeigen
52.  @toEditor: sample
53.
54.  # Script pausieren, damit der User das zu isolierende Target markieren kann
55.  @userInteraction: "Target markieren: " + pinyin$ + " - Pos.: " + pos$ + " -
    Length: " + len$

```

```

56.
57. # Start und Ende des markierten Targets ermitteln
58. targetStart = Get start of selection
59. targetEnd = Get end of selection
60.
61. @toShell
62.
63. # extract and store the selected sound sample (target)
64. target = Extract part: targetStart, targetEnd, "rectangular", 1, "no"
65. selectObject: target
66. Save as WAV file: fpTarget$
67. targetStart$ = string$: targetStart
68. targetEnd$ = string$: targetEnd
69. @log: target_id$ + " - " + "Target: " + fpTarget$ + "|" + targetStart$ + "|" +
targetEnd$
70.
71. # analyze sound samples (target) and store retrieved values
72. @analyzeSyllable: target, "target"
73.
74. @log: target_id$ + " - " + "Target stored as: " + fpTarget$)
75.
76. # open target sound in editor
77. @toEditor: target
78.
79. # pause the script to mark the first syllable boundary
80. @userInteraction: "Markiere die erste Silbengrenze"
81.
82. # store first syllable boundary
83. boundary1 = Get cursor
84.
85. # pause the script to mark the second syllable boundary
86. @userInteraction: "Markiere die zweite Silbengrenze"
87.
88. # store second syllable boundary
89. boundary2 = Get cursor
90.
91. boundary1$ = string$: boundary1
92. boundary2$ = string$: boundary2
93.
94. @log: target_id$ + " - " + "Syllable boundaries: " + boundary1$ + "|" +
boundary2$
95.
96. @toShell
97.
98. @getStartEnd: target
99.
100. # cut out the three sound samples (syllables)
101. selectObject: target
102. syllable1 = Extract part: getStartEnd.start, boundary1, "rectangular", 1, "no"
103. selectObject: target
104. syllable2 = Extract part: boundary1, boundary2, "rectangular", 1, "no"
105. selectObject: target
106. syllable3 = Extract part: boundary2, getStartEnd.end, "rectangular", 1, "no"
107.
108. # store the three sound samples (syllables)
109. selectObject: syllable1
110. Save as WAV file: fpSyllable1$

```

```

111. selectObject: syllable2
112. Save as WAV file: fpSyllable2$
113. selectObject: syllable3
114. Save as WAV file: fpSyllable3$
115.
116. # analyze sound samples and store retrieved values
117. @analyzeSyllable: syllable1, "s1"
118. @analyzeSyllable: syllable2, "s2"
119. @analyzeSyllable: syllable3, "s3"
120.
121. # write target_id to targets-done-file
122. appendFileLine: fpTargetsDone$, target_id$
123.
124. # Alle Objekte schließen
125. @close: sample
126. @close: target
127. @close: syllable1
128. @close: syllable2
129. @close: syllable3
130.
131.
132. #
133. # Prozeduren
134. #
135.
136.
137. #
138. # analyzes a sound object for pitch data, stores pitch listing
139. #
140. procedure analyzeSyllable: .object, .idx$
141.
142. # prepare variables
143. #   fpPitchListing$ = truncpathTarget$ + "_" + .idx$ + "_pitch" + ".csv"
144. #   fpTimesListing$ = truncpathTarget$ + "_" + .idx$ + "_times" + ".csv"
145. fpPitchListing$ = truncpathTarget$ + "_" + .idx$ + "_data" + ".csv"
146. fpPicture$ = truncpathTarget$ + "_" + .idx$ + "_picture" + ".png"
147.
148. # get parameters from sound object for pitch range
149. # cf. Looze (2010)
150. select .object
151. To Pitch... 0.01 60 700
152. lower = Get quantile... 0 0 0.25 Hertz
153. mmin = 0.75 * lower
154. upper = Get quantile... 0 0 0.75 Hertz
155. mmax = 2 * upper
156. Remove
157.
158. # get pitch and intensity objects
159. select .object
160. .objectPitch = To Pitch: 0.001, mmin, mmax
161. select .object
162. .objectIntensity = To Intensity: mmin, 0.001
163.
164. # loop through 0.01-sec-frames and get time, pitch and intensity
165. select .object
166. @getStartEnd: .object
167. tmax = getStartEnd.end

```

```

168. tmin = getStartEnd.start
169. for i to (tmax-tmin)/0.01
170. time = tmin + i * 0.01
171. selectObject: .objectPitch
172. pitch = Get value at time: time, "Hertz", "linear"
173. selectObject: .objectIntensity
174. intensity = Get value at time: time, "cubic"
175. listItem$ = fixed$(time, 2) + "|" + fixed$(pitch, 3) + "|" + fixed$(intensity,
3)
176. appendFileLine: fpPitchListing$, listItem$
177. endfor
178.
179. # get min/max from pitch object
180. select .objectPitch
181. .min = Get minimum: 0, 0, "Hertz", "none"
182. .max = Get maximum: 0, 0, "Hertz", "none"
183. .minPos = Get time of minimum: 0, 0, "Hertz", "none"
184. .maxPos = Get time of maximum: 0, 0, "Hertz", "none"
185.
186. # # get and store pitch/time listing
187. # pitchListing# = List values in all frames... Hertz
188. # timesListing# = List all frame times
189.
190. # get min/max from intensity object
191. select .objectIntensity
192. .minI = Get minimum: 0, 0, "parabolic"
193. .maxI = Get maximum: 0, 0, "parabolic"
194. .minPosI = Get time of minimum: 0, 0, "parabolic"
195. .maxPosI = Get time of maximum: 0, 0, "parabolic"
196.
197. # draw pitch
198. select .objectPitch
199. Solid line
200. Line width... 3
201. Blue
202. Draw: 0, 0, mmin, mmax, "yes"
203.
204. # draw intensity
205. select .objectIntensity
206. Dashed line
207. Line width... 3
208. Green
209. Draw: 0, 0, 50, 80, "no"
210.
211. @printPicture: fpPicture$
212.
213. # prepare retrieved data
214. .min$ = string$: .min
215. .max$ = string$: .max
216. .minPos$ = string$: .minPos
217. .maxPos$ = string$: .maxPos
218. mmin$ = string$: mmin
219. mmax$ = string$: mmax
220. duration$ = string$: getStartEnd.duration
221.
222. .csv$ = target_id$ + "|" + .idx$ + "|" + duration$ + "|" + mmin$ + "|" + mmax$ +
|" + .maxPos$ + "|" + .max$ + "|" + .minPos$ + "|" + .min$ + "|" + date$()

```

```
223.
224. # store data to files
225. appendFileLine: fpData$, .csv$
226. appendFileLine: fpTargetData$, .csv$
227. # time/pitch/intensity stored already above in loop
228.
229. @close: .objectIntensity
230. @close: .objectPitch
231. endproc
232.
233.
234. #
235. # pauses the script for interaction or for exiting
236. #
237. procedure userInteraction: .message$
238. beginPause: .message$
239. .action = endPause: "Exit", "Abbrechen", "Weiter", 3
240. if .action == 1
241. @log: target_id$ + " - " + "System script abort by user!"
242. @phpStop
243. exitScript()
244. elsif .action == 2
245. @log: target_id$ + " - " + "Praat script abort by user!"
246. exitScript()
247. endif
248. endproc
249.
250.
251. #
252. # get start and end time of a sound object
253. #
254. procedure getStartEnd: .object
255. @toEditor: .object
256. .soundinfo$ = Sound info
257. .start = extractNumber (.soundinfo$, "Start time:")
258. .end = extractNumber (.soundinfo$, "End time:")
259. .duration = .end - .start
260. @toShell
261. endproc
262.
263. #
264. # select a sound object and switch to the editor
265. #
266. procedure toEditor: .object
267. # Sample im Editor aktivieren
268. selectObject: .object
269. View & Edit
270. editor: .object
271. endproc
272.
273. #
274. # switch back to the shell (Praat objects window)
275. #
276. procedure toShell
277. Close
278. endeditor
279. endproc
```

```
280.
281. #
282. # save picture content to file and clean it
283. #
284. procedure printPicture: .fp$
285. Save as 600-dpi PNG file: .fp$
286. Erase all
287. endproc
288.
289. #
290. # close a given object and remove it from Praat objects window
291. #
292. procedure close: .object
293. selectObject: .object
294. Remove
295. endproc
296.
297. #
298. # write given content to a log file
299. #
300. procedure log: .content$
301. appendFileLine("444.log", date$() + ": " + .content$)
302. endproc
303.
304. #
305. # create a file as flag for PHP to stop execution
306. #
307. procedure phpStop:
308. appendFileLine(fpPhpFlag$, "System script abort by user!")
309. endproc
```