Blender 2.79 instructions for exporting animated skeletal mesh with clothes using manual bastioni lab 1.6

1. Open blender
2. Add manual bastioni character after choosing the required phenotype (use IK no muscles), cycles, lights if needed.
3. Switch off Subsurf
4. Select shape and face features using sliders
5. Export and backup all textures and json files for the character
6. Select armature and put it into rest mode
7. Add clothes to model
8. Fit clothes without using proxy
9. After all clothes are fitted time to add weights
10. Select cloth or hair to add weight
11. Attach it to parent skeleton with empty groups
12. Select character mesh and put in weight paint mode
13. Select cloth/hair and put in weight paint mode
14. Select character mesh and armature then shift select the cloth/hair and transfer weights
15. Use nearest face or projected face depending on the cloth
16. Choose by name and all layers for source and destination and click transfer weights again
17. After transferring weights go back to pose mode and check the fit by moving the armature
18. Repeat for additional clothes and hair
19. Dangling and swinging parts are attached to separate armature created specifically for it having the same root as original mesh
20. Go back to Rest Position
21. Do not Delete Subdivision
22. Make a backup of blend file
23. Finalize the character
24. Add breath shape keys
25. Change metric system and scale to 0.01
26. Scale everything by 100   
    (or)   
    skip step 26 and 27 but during export enter scale as 100 there this is not recommended though. Whichever process you follow for your 1st character must remain same for the others too
27. Ctrl a to apply scaling and rotation
28. Remove hidden vertices
29. Among the clothes and hair there remove the vertex groups which you do not think should be affecting the weights. Example most hair and glasses will only require weights from head and struct\_hd
30. Merge all clothes and body
31. Rename character to main\_root
32. Give a name for the mesh inside the object main\_root\_character
33. Export using fbx do not apply modifiers/leaf bones/baked animation/
34. Choose -y forward
35. Only selected objects not all (skeleton and joined mesh)
36. Use face smoothing

For Creating Lip Synced Audio

1. Download and install the latest version of version of Papagayo (I use 2.0b1)
2. Export your audio as wave [WAV (Microsoft) signed 16-bit PCM in audacity is what I use]
3. Open Papagayo and drop your wave file in and match the audio to the phonemes after typing in the spoken text
4. Set FPS as 60 in the text box to the top right
5. After you are satisfied with sync Export with Anime Studio Option and save the file with .dat extension
6. Install python 3
7. Use pip to install pyperclip
   1. pip install pyperclip
8. Change directory in the prompt to the VoiceTools Folder and run the program using
   1. python fblahTools.py
9. Open the dat file you exported
10. Click Convert
11. Papagayo’s phonemes are replaced by names usable with MBLab
12. Click Make one line to reduced the data to a single line
13. Click copy to move the data to clipboard
14. Open the fblah MBLab study project
15. Create an instance of Face and Voice manager in a map by dragging the actor in
16. Select that actor and look at the details panel
17. Under import animation
    1. paste clipboard contents in Face Data Stream
    2. Click *Prepare Input From Voice Tools* Button
    3. notice the Output Face Time Data gets populated
    4. right click the array and click copy
18. It is good to keep all audio lip syncs in a data table. In the test project the table is called *AudioDataTable* (Content/fBlah/Common/BP/Data) It uses struct called AudioFaceAnimation
19. Open it and add an entry to it
20. Paste the contents of clipboard to FaceTimeData
21. Import your audio wave into the project and create cue for it (Right Click->Create Sound Cue)
22. Apply that cue to the Audio field.
23. Set the Row Name with a name you want to reference it with later
24. Check TestMap2 for an example on how to play audio with lip sync. (Content/fBlah/TestMaps). The level blueprint has all the information you need.