

# Digital Semiology Manual

3.....	Short overview
3.....	Getting Started
3.....	Installation (Windows)
7.....	DIGITAL SEMIOLOGY: GENERAL ASPECTS
7.....	Digital semiology graphical user interface:
8.....	Writing GUI
8.....	Selecting GUI
9.....	The role of IPython console in DS
10.....	Presenting error message
10.....	Files
12.....	DS workflow
12.....	OPERATING VIDEO PLAYER
12.....	Video Player Overview
13.....	Installing Video Player
13.....	Starting Video Player
18.....	Using Video Player
18.....	Timing Indication
20.....	Closing Video Player
21.....	INTRODUCTION PART OF DS
21.....	Name of the User
22.....	Ictal Episode Code
22.....	Overwrite or Edit
23.....	General View or Add-on Mode
24.....	Machine Decisions: Yes or No?
25.....	Video Player: Yes or No?
266.....	Looking at EEG, when interpreting video
26.....	Ictal Episode Date
27.....	Timings of Ictal Episode Start
27.....	Baseline Behavior?
28.....	PNES?
29.....	Whether ictal episode started from sleep?
29.....	Cognitive/motor Testing During Ictal Episode?
30.....	Hierarchical description of ictal semiology

31.....	EVENTS
31.....	Events: general aspects
31.....	Events: add-on mode
32.....	Events: general view mode
34.....	Abbreviations and terms used in DS
34.....	BEHAVIORS
35.....	SIMPLE MOTOR BEHAVIORS
36.....	Tonic and dystonic behaviors
39.....	Fencer posturing
42.....	Clonic behaviors
44.....	Epileptic spasm
45.....	AUTOMATISMS
49.....	HYPERKINETIC BEHAVIORS
50.....	GENERALIZED TONIC CLONIC SEIZURES
52.....	AUTONOMIC BEHAVIORS
53.....	EYE/EYELID MOVEMENT BEHAVIORS
58.....	VOICE BEHAVIORS
59.....	DIALEPTIC BEHAVIORS
60.....	AURA REPORTING BEHAVIORS
61.....	Body parts in aura reporting behaviors
62.....	Aura verbal report (certain)
66.....	OTHER BEHAVIORS
67.....	TRIGGER
67.....	SEMILOGIC EXTRAPOLATION FUNCTIONS ("MACHINE DECISIONS")
68.....	Focality criteria
68.....	Motor jacksonian march criteria
69.....	PNES criteria
69.....	SUDEP risk criteria
69.....	EDITING ICTAL EPISODE
69.....	Event removal
70.....	Editing reported event
71.....	Editing introduction
72.....	Edit once again
73.....	Postictal
74.....	Writing final comment

75.....	DS exit
75.....	The nuances of ictal episode editing in General view mode
77.....	DIGITAL SEMIOLOGY REPORT
77.....	DS Report: Overview
77.....	Short description of ictal episode
77.....	Detailed description of ictal episode
78.....	Statements regarding postictal state
78.....	Final comment
78.....	Writing free text comments in DS
80.....	EXIT FROM DS
81.....	THE PROGRAM UNPACK_ICTAL_EPISODE
83.....	THE PROGRAM DS_VERSION_ADAPTER

## Short overview

Digital Semiology (DS) is a Python based research tool for interpreting, encoding and reporting epilepsy ictal semiology, recorded by video. DS cannot be used for detecting seizures and it assumes that seizures were detected by other means: manually or by algorithms. DS does not take into account EEG and, therefore, cannot be used solely for seizure classification. However, DS, if combined with EEG interpretation, can assist to classify seizures. The output of DS is a report saved in binary file, which can be opened using Python based program: unpack\_ictal\_episode.

In addition to basic Python, DS requires installation of WxPython. The program unpack\_ictal\_episode requires installation of numpy and MatPlotLib modules.

## Getting Started

### Installation (Windows)

1. Install anaconda from <https://www.anaconda.com>
2. Install WxPython.

Write in Search: *anaconda prompt*. Open the prompt. In the prompt write *conda install wxpython* and press Enter.

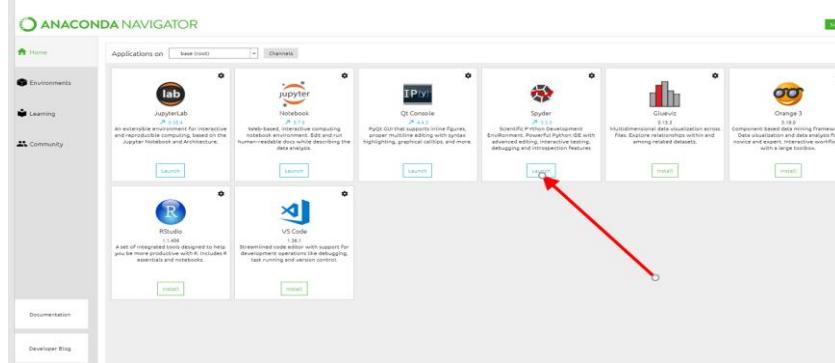
3. Installation Digital Semiology and unpack\_ictal\_episode.

Place files Digital\_Semiology.py and unpack\_ictal\_episode.py into folder:

.spyder-py3. (usually: This PC > Windows(C:) > Users > name of the computer user > .spyder-py3)

## Start Digital Semiology

1. Write in Search: *anaconda navigator*.
2. Left double click on Anaconda Navigator App.
3. Left double click on the Launch button of Spyder – this will open Spyder environment.



The details about Spyder can be found in <https://www.spyder-ide.org>

4. Check whether *Digital\_Semiology.py* is listed in the Editor window.

```
try:
    event_start=int(eventstart)
except:
    eventstart=""
if len (eventstart) !=6 or int(eventstart[0:2])<0 or
    int(eventstart[4:])<0 or int(eventstart[4:])-int(eventstart[0:2])>6:
    print ('''\a\nSOMETHING WRONG! PLEASE TRY AGAIN''')
#Step 5 - Introduction timing of the end of ictal
eventend=""
while len (eventend) !=6 or int(eventend[0:2])<0 or
    int(eventend[4:])<0 or int(eventend[4:])-int(eventend[0:2])>6:
    print("\n",nm," ",end="")
    eventend=input('Write the time of ictal episode')
try:
```

You can use the Browse tabs button in the Editor window – left single click.

```

Spyder (Python 3.7)
File Edit Search Source Run Debug Consoles Projects Tools View
Editor - C:\Users\maric\spyder-py3\Digital_Semiology_15.07.2019.py
Digital_Semiology_15.07.2019.py hellow.py
42     try:
43         event_start=int(eventstart)
44     except:
45         eventstart=''
46     if len (eventstart) !=6 or int(eventstart[0:2])
47         or int(eventstart[4:])>0 or int(eventstart[4
48             print ('''\a\nSOMETHING WRONG! PLEASE TR
49 #-----
50 #Step 5 - Introduction timing of the end of ictal
51 eventend=''
52 while len (eventend) !=6 or int(eventend[0:2])<0 or
53     or int(eventend[4:])<0 or int(eventend[4:])>
54     print("\n",nm,",",end='')
55     eventend=input('Write the time of ictal episode
56     try:

```

5. If not listed, left single click on the Open File button in the Spyder menu (black open folder in the left upper corner).

```

Spyder (Python 3.7)
File Edit Search Source Run Debug Consoles Projects Tools View
Editor - C:\Users\maric\spyder-py3\Digital_Semiology_15.07.2019.py
Digital_Semiology_15.07.2019.py hellow.py
42     try:
43         event_start=int(eventstart)
44     except:
45         eventstart=''
46     if len (eventstart) !=6 or int(eventstart[0:2])
47         or int(eventstart[4:])<0 or int(eventstart[4
48             print ('''\a\nSOMETHING WRONG! PLEASE TR
49 #-----
50 #Step 5 - Introduction timing of the end of ictal
51 eventend=''
52 while len (eventend) !=6 or int(eventend[0:2])<0 or
53     or int(eventend[4:])<0 or int(eventend[4:])>
54     print("\n",nm,",",end='')
55     eventend=input('Write the time of ictal episode
56     try:

```

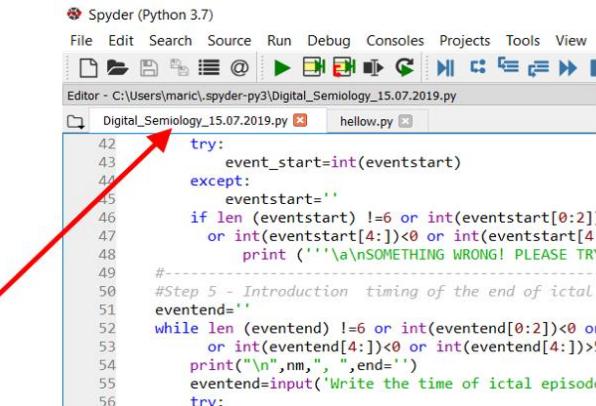
After the directory is open, left double click on the Digital\_Semiology.py.

```

Spyder (Python 3.7)
File Edit Search Source Run Debug Consoles Projects Tools View Help
Editor - C:\Users\maric\spyder-py3\Digital_Semiology_15.07.2019.py
Digital_Semiology_15.07.2019.py hellow.py
42     try:
43         event_
44     except:
45         events_
46         if len (events_
47             or int(events_
48                 print(
49 #-----
50 #Step 5 - Intro
51 eventend=''
52 while len (eventend)
53     or int(eventend)
54     print("\n")
55     eventend=
56     try:
57         event_
58     except:
59         events_
60         if len (events_
61             or int(events_
62                 print(
63 #-----
64 #Step 6 - Com
if int(eventst

```

6. Left single click on the Digital\_Semiology.py button in the upper part of the Editor window – the background of the button converted to white.

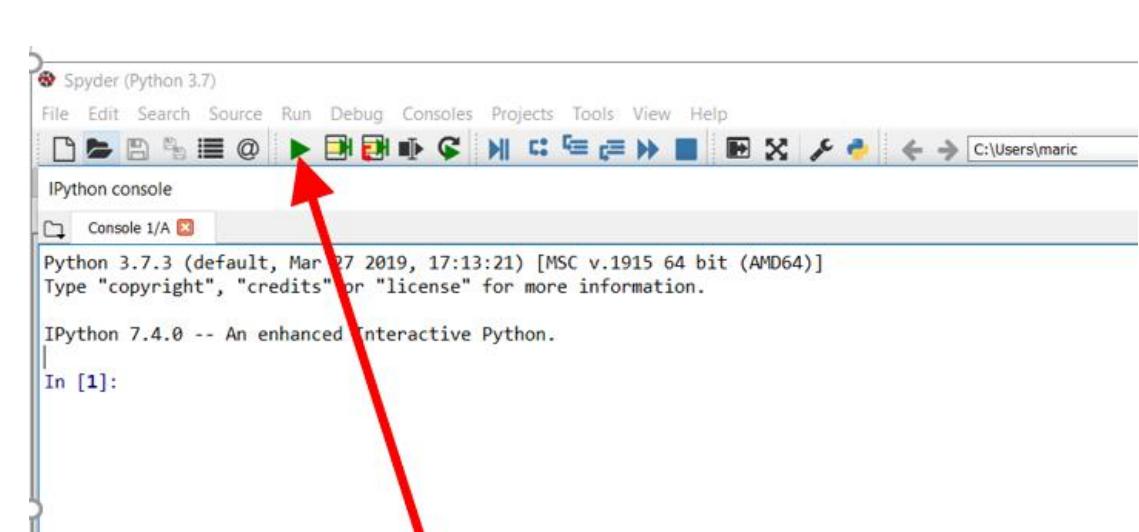


```

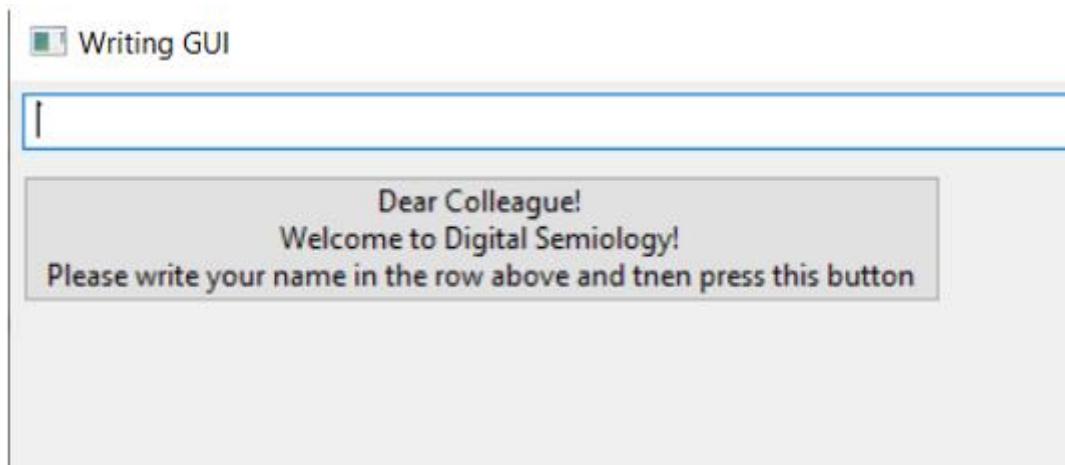
Spyder (Python 3.7)
File Edit Search Source Run Debug Consoles Projects Tools View
Editor - C:\Users\maric\spyder-py3\Digital_Semiology_15.07.2019.py hellow.py
Digital_Semiology_15.07.2019.py
try:
    event_start=int(eventstart)
except:
    eventstart=''
if len (eventstart) !=6 or int(eventstart[0:2])<0 or int(eventstart[4:6])>9:
    print ('''\a\SOMETHING WRONG! PLEASE TRY AGAIN''')
#Step 5 - Introduction timing of the end of ictal
eventend=''
while len (eventend) !=6 or int(eventend[0:2])<0 or int(eventend[4:6])>9:
    print("\n",nm," ",end='')
eventend=input('Write the time of ictal episode')
try:

```

7. Since user can follow the DS performance through IPython console, we recommend enlarging IPython console by dragging its edges (simultaneously pressing the left mouse).



On the screen appears GUI:

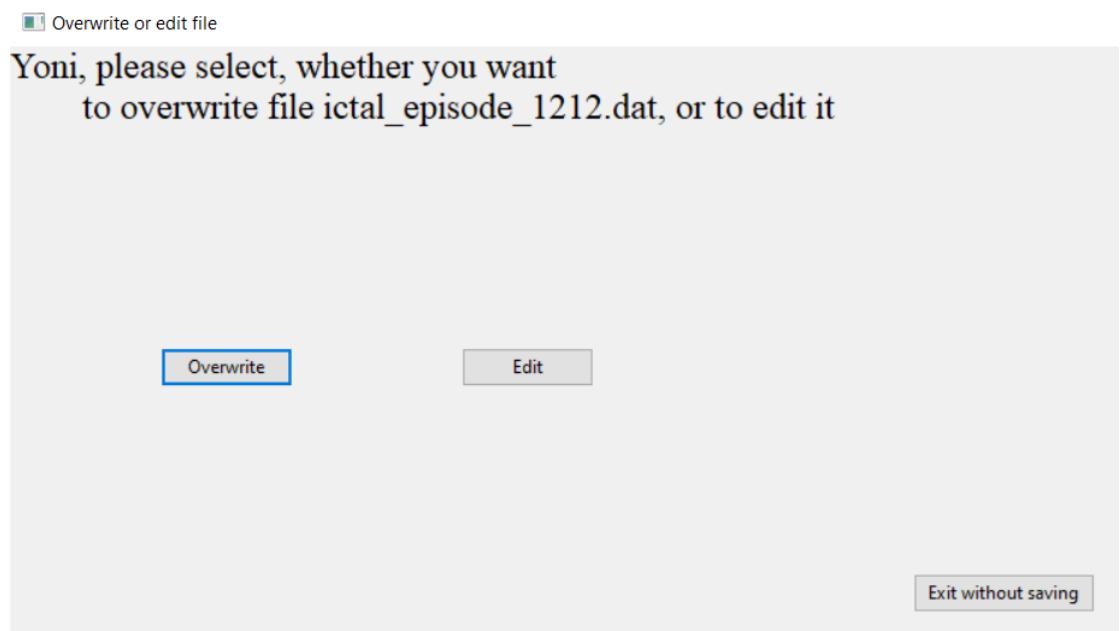


## DIGITAL SEMIOLOGY: GENERAL ASPECTS

Digital semiology graphical user interface:

DS uses two types of GUI:

- 1) Writing GUI - an example you can see right above.
- 2) Selecting GUI – an example is right below:



When DS generates GUI, the second Spyder symbol appears on the lower part of the screen.

The screenshot shows the Spyder Python 3.7 IDE. In the top left, there's a toolbar with various icons. The main area has a code editor titled 'hellow.py' containing Python code. To the right of the code editor is a 'Console I/A' tab showing a command-line interface where a user can input commands like 'Marik , please define the chance that this ictal episode is PNPNNE (pathologic non-psychogenic non-epileptic) in a scale between 0 and 2 and then press Enter'. Below the code editor is a 'History log' tab showing a list of runfiles and their execution times. At the bottom of the interface, there's a status bar displaying file permissions, encoding, line number, column, memory usage, and date.

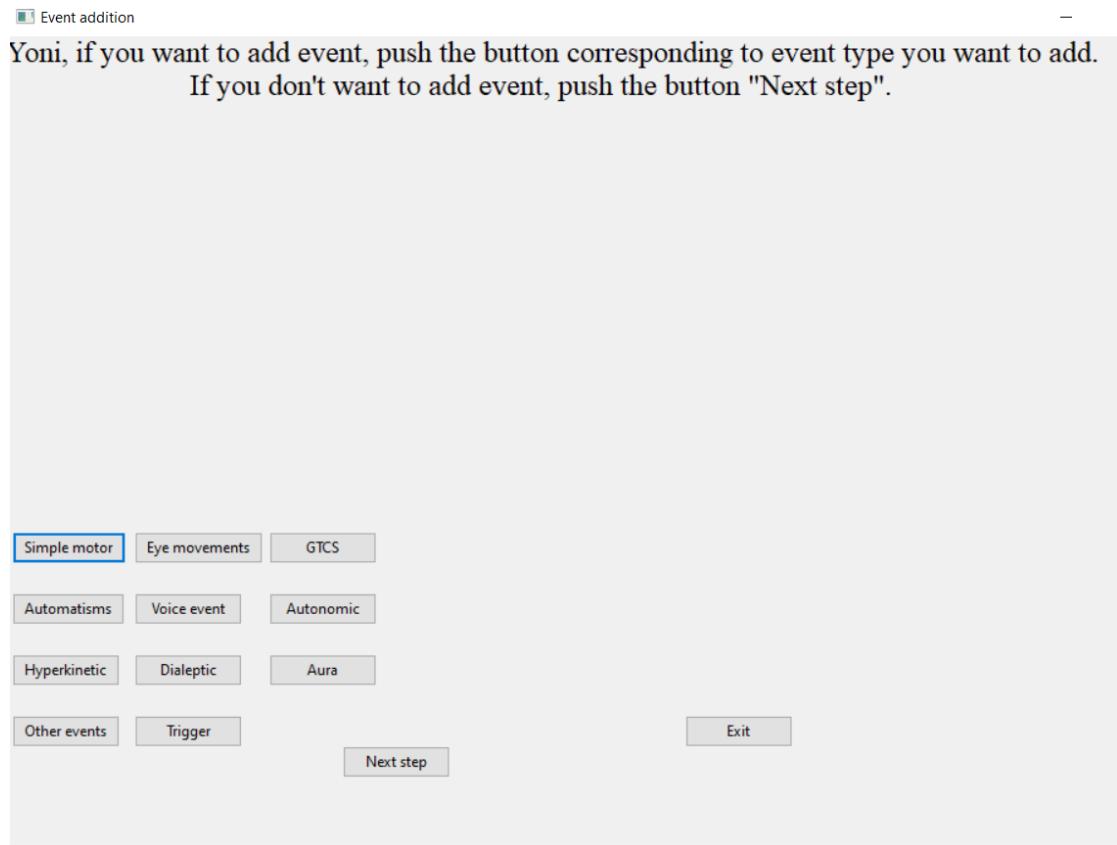
GUI either automatically appears on the screen or the user should click on the second Spyder symbol to make GUI visible.

## Writing GUI

Writing GUI contains two parts: the row for writing and the button with instructions. User is expected to write something in the row (according to instructions on the button) and then press the button, which is located right below the row.

## Selecting GUI

Selecting GUI presents several options, each corresponds to individual button. User can see appropriate instructions in the upper part of Selecting GUI. Often, Selecting GUI allows only one option selection and after pressing any button, the GUI disappears, and DS continues to the next step. In some cases, it is possible to press sequentially on several buttons, sometimes, however, not all combinations are allowed. Often the Selecting GUI has the button "Next step"; clicking on this button quits the GUI and DS continues to the next step.



With appearance of some Selecting GUIs, the short explanations of terminology appear in IPython Console (see the paragraph "Terminology explanation").

### The role of IPython console in DS

In DS, IPython console serves for 5 purposes:

1. Displaying the user-software (DS) communication history during the session (see the paragraph "User-software communication history").
2. Presenting some explanations regarding the terminology used in GUIs (see the paragraph "Terminology explanation").
3. In the case when DS opens existing file, to see its content (see the paragraph "Files").
4. To view the incomplete (or complete) report during or at the end of the process of the ictal episode description (see the paragraph "DS Report").
5. One of the ways to write the comments (see the paragraph "Writing comments").
6. Presenting error message.

### User-software (DS) communication history

All steps performed by DS and the user during encoding are logged in realtime in the IPython console and later saved at the end of the report file. The steps performed by DS are labeled SOFTWARE: ... , and the steps of the user are labeled USER: ... All steps

are presented in the chronological order and the timings of execution are also presented in IPython console.

Here is an example of User/Software communication history:

```
21:58:17
Software:
Dear Colleague!
Welcome to Digital Semiology!
Please write your name in the row above and then press this button

21:58:34
User typed:
Yoni

21:58:34
Software:
Yoni, please, write the CODE of ictal episode in the window above and then press this button

21:58:42
User typed:
1212

21:58:42
Software:
Yoni, please select, whether you want
    to overwrite file ictal_episode_1212.dat, or to edit it
```

In addition to communication timings, the date of the session is displayed in IPython at the start of Digital Semiology.

```
Software:
Today's date: 2020-09-23
```

The User/Software communication is saved in the variable `software_user_dialogue`. This variable is saved in the binary file (see the paragraph "Files").

### Presenting error message

DS presents error message in IPython console. The content of error messages can be different. Here is an example:

```
Software:
SOMETHING WRONG! PLEASE TRY AGAIN!
```

### Files

DS creates, updates or reads several files.

#### **ictal\_episode\_ictal-episode-code.dat file (binary)**

The main fail, which is created or updated is .dat binary file. This file includes three variables: 1) `ds_starter`; 2) `ictus`; 3) `software_user_dialogue`.

`ds_starter` is a variable, which includes general information about ictal episode (not including details of events)

*ictus* is a variable, which includes numerically encoded information about events in ictal episode.

*report* is another variable, which verbally describes an ictal episode (More details in the paragraph DS report).

*software\_user\_dialogue* is a variable, which includes information about all steps that user and software took during interpretation of ictal episode, including the date of the interpretation and timings of every user's or software's step. If user interpreted an ictal episode in several sessions, each new DS start is labeled by the date.

The name of .dat binary file is created as: ictal\_episode\_ictal-episode-code.dat. For example, if ictal episode code is 1212, the filename will be ictal\_episode\_1212.dat.

If .dat binary file of the ictal episode (with the code of this ictal episode) doesn't exist in the same directory as DS, DS will create this file. If exists – the user can choose whether overwrite this file or to edit it. In the case, when editing option was selected, all three variables of the file are updated. If overwriting option is selected, ictus and report are overwritten, while software\_user\_dialogue is updated. The editing and overwriting don't be executed, if during the exit from the program the user selects option "do not save".

### **ictal\_episode\_ictal-episode-code\_draft.dat file (binary)**

This is a back-up file, which is updated at the end of every event interpreting. In the case of incorrect exit from the program (e.g. forced shutdown), this file remains in the directory of DS. In the case of correct exit from the program, this file is deleted from the directory.

### **ictal\_episode-code.txt file**

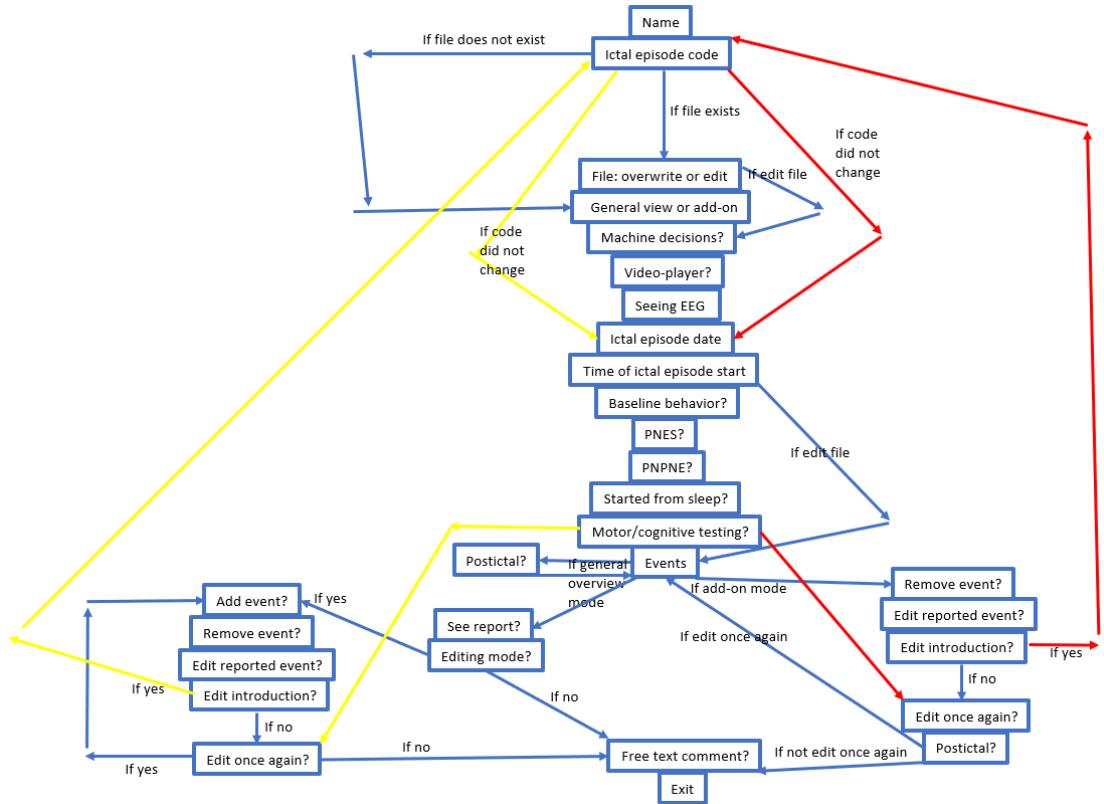
The content of .dat binary file can be written into .txt file using python code: unpack\_ictal\_episode (see separate paragraph about this code). The name of this .txt file is also composed as: ictal\_episode\_ictal-episode-code.txt. If ictal episode code is 1212, then the name of this file will be ictal\_episode\_1212.txt. We recommend placing the DS code and unpack\_ictal\_episode code in the same directory. In that case also .dat binary file and .txt file will be in the same directory.

### **video-player\_go\_no\_go.txt and intercode.txt**

These files serve for interactions between DS code and video-player code. They are created by DS and are in the same directory.

All files associated with DS are in the same directory with the DS code, except of the files, associated with video-player, which are discussed in the paragraph "Videoplayer".

## DS workflow



It is practical to Divide DS workflow to three parts:

1. Introduction – everything before Events
2. Events
3. Editing ictal episode – everything after Events (not to miss with editing existing file).

## OPERATING VIDEO PLAYER

### Video Player Overview

The important function of DS is timing labelling of events during ictal episode. DS allows to do this in one of two ways: 1) manually and 2) using a video player interacting with DS. Video player has also the audio features, so it is more correctly to name it video-audio player.

The code supporting the video player is a separate python code: `playing_a_video.py`, which should be placed to the same directory with DS code. The communication between these two codes (`DigitalSemiology.py` and `playin_a_video.py`) is done by writing and reading two text files by these codes:

1. `video_player_go_no_go.txt`
2. `intercode.txt`

Video player uses video files of the type .wmv. These files should not necessarily be in the same file with DS. The path to .wmv-file should be pasted to appropriate place in the playing\_a\_video.py code.

### [Installing Video Player](#)

#### *Installing new packages in anaconda environment*

1. Open Anaconda.
2. Select Environments in the left-hand pane (below home).
3. Just to the right of where you selected and below the "search environments" bar, you should see "base(root)" - Click on it.
4. A triangle pointing right should appear, click on it and then select "open terminal".
5. Use the regular pip install command here. There is no need to point to an environment/ path.

#### *What to install*

1. Install wxPython: pip install wxPython==4.0.7 (if not installed previously)
2. Install MplayerCtrl: pip install MplayerCtrl
3. Download MPlayer zip file, unzip after downloading, save in same file as other python codes.  
**OR:** go to site: <http://www.mplayerhq.hu/design7/news.html>, download: MPlayer 1.4 HTTP (xz, 15 MB), unzip after downloading, save in same file as other python codes.
4. Place code playing\_a\_video in the same directory as other python codes.

### [Starting Video Player](#)

#### *Setting the path*

At first the user should paste the path to .wmv-file in the code playing\_a\_video.py.

```

195 while read_video== :
196     text_file=open("video-player_go_no_go.txt", "r", encoding='utf-8')
197     read_video=text_file.read(10)
198     text_file.close()
199     print('mobile perpetum')
200 if read_video=='0':
201     quit()
202 if read_video=='1':
203     # CHOOSE VIDEO HERE
204     myVideo=r"C:\Users\maric\Downloads\marik_newer\"+ictal_episode_code+".wmv"
205
206 app = wx.App(0)
207 frame = wx.Frame(None, size=(640, 480))
208 panel = TestPanel(frame)
209 frame.Show()
210 app.MainLoop()
211
212

```

The path should be pasted to line 204. In this example the path is:

C:\Users\maric\Downloads\marik\_newer\\

In general form line 204 can be presented as:

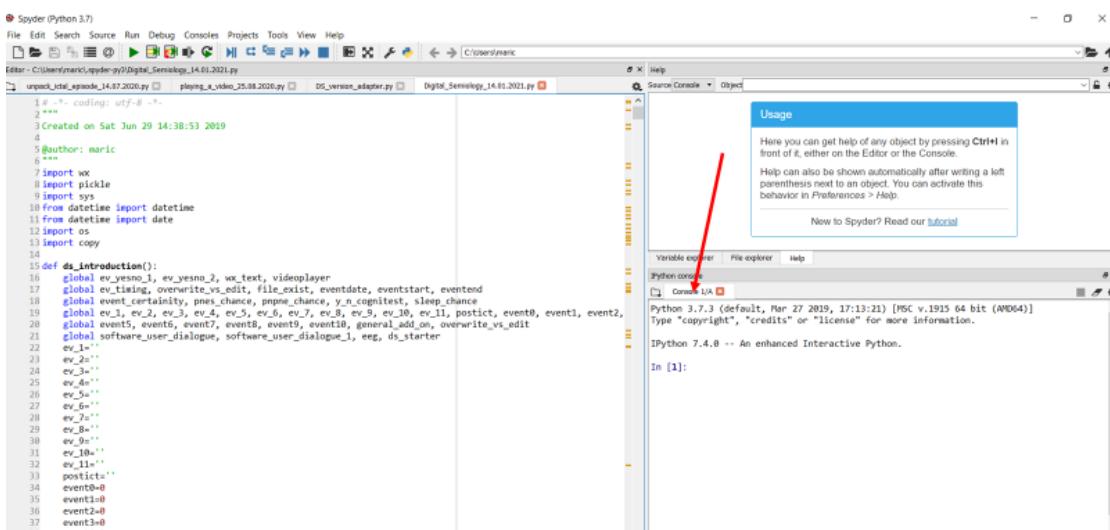
myVideo=r"path"+ictal\_episode\_code+".wmv"

It is prudent to use all the time the same directory where the .wmv-files (video files) are placed, so that it will be no need to reset the path.

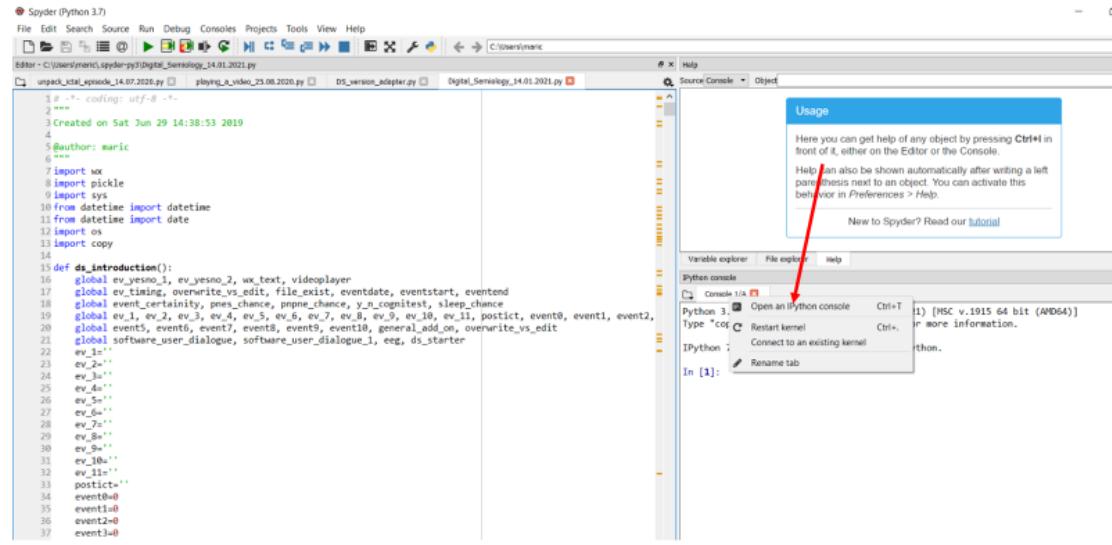
### *Opening video player*

It is important to open video player in a terminal separate from DS code. It can be achieved by double click on the playing\_a\_video.py in the directory, where this file is located.

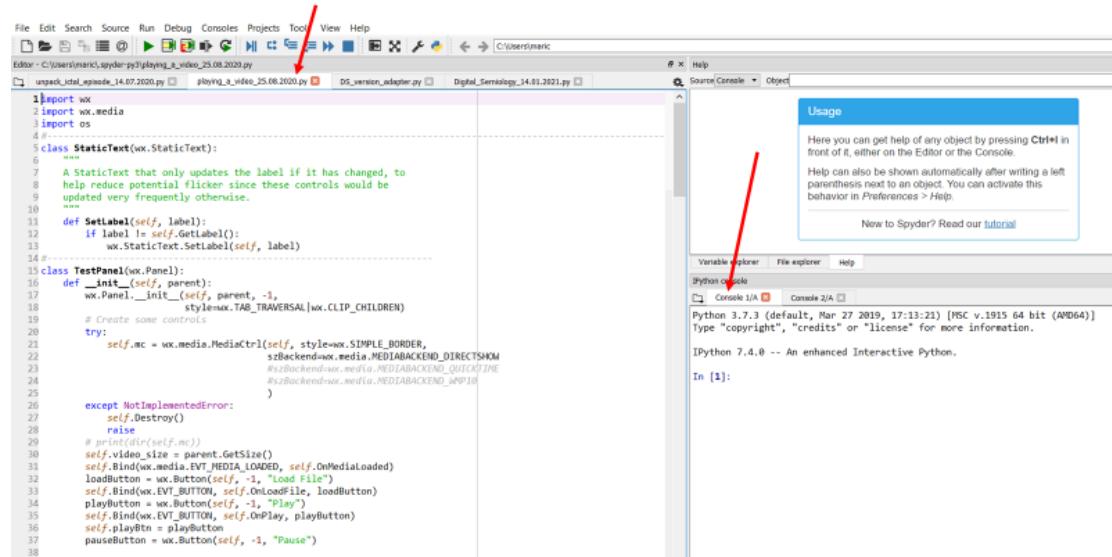
Another way is using user interface of Spyder. At first, the user should make single left click to the console window:



The small window will open. The user should select the uppermost option "open the IPython console":



This will open the second console. User should make single click (left) on one of consoles window and to start run plaing\_a\_video code.



In the console running row "perpetum mobile" will appear.

The screenshot shows the Spyder Python IDE interface. The code editor on the left contains Python code for a media player application, including imports for wx and wx.media, and definitions for StaticText and TestPanel classes. The console window at the bottom displays a series of 'perpetum mobile' entries. A help panel on the right provides information on using the Ctrl+H hotkey for help and describes how help is activated by pressing the left parenthesis key.

```
1 import wx
2 import wx.media
3 import os
4 # ...
5 class StaticText(wx.StaticText):
6     """
7         A StaticText that only updates the label if it has changed, to
8         help reduce potential flicker since these controls would be
9         updated very frequently otherwise.
10    """
11    def SetLabel(self, label):
12        if label != self._getLabel():
13            wx.StaticText.SetLabel(self, label)
14    # ...
15 class TestPanel(wx.Panel):
16    def __init__(self, parent):
17        wx.Panel.__init__(self, parent, -1,
18                          style=wx.TAB_TRAVERSAL|wx.CLIP_CHILDREN)
19    # Create some controls
20    try:
21        self.mc = wx.media.MediaCtrl(self, style=wx.SIMPLE_BORDER,
22                                     szBackend=wx.media.MEDIABACKEND_DIRECTSHOW
23                                     #szBackend=wx.media.MEDIABACKEND_QUICKTIME
24                                     #szBackend=wx.media.MEDIABACKEND_WMP10
25                                     )
26    except NotImplementedError:
27        self.Destroy()
28        raise
29    # print(dir(self.mc))
30    self.video_size = parent.GetSize()
31    self.Bind(wx.media.EVT_MEDIA_LOADED, self.OnMediaLoaded)
32    loadButton = wx.Button(self, -1, "Load file")
33    self.Bind(wx.EVT_BUTTON, self.Onloadfile, loadButton)
34    playButton = wx.Button(self, -1, "Play")
35    self.Bind(wx.EVT_BUTTON, self.Onplay, playButton)
36    self.playBtn = playButton
37    pauseButton = wx.Button(self, -1, "Pause")
38    eventButton = wx.Button(self, -1, "Event")
39
```

Next, user should make single left click on the second consoles window and run digital semiology code:

A screenshot of the Spyder Python IDE interface. The top menu bar includes File, Edit, Search, Source, Run, Debug, Consoles, Projects, Tools, View, Help. The toolbar has icons for file operations like Open, Save, Run, Stop, and Help. The left sidebar shows project files: unpack\_itself\_14.07.2020.py, playing\_a\_video\_25.08.2020.py, DS\_version\_adapter.py, and Digital\_Semiology\_14.01.2021.py. The main code editor window displays a Python script with numerous comments starting with '#'. A red arrow points from the top right towards the Help panel. The Help panel title is 'Usage' with a sub-instruction: 'Here you can get help of any object by pressing Ctrl+H in front of it, either on the Editor or the Console.' It also says 'Help can also be shown automatically after writing a left parenthesis next to an object. You can activate this behavior in Preferences > Help.' Below this is a link 'New to Spyder? Read our tutorial'. The bottom right shows the IPython console with tabs 'Console 1/A' and 'Console 2/A', both of which are selected. The console output shows Python version 3.7.3 and 4.7.0 respectively, along with copyright and license information.

The following text will appear in the IPython console:

```

1 # -*- coding: utf-8 -*-
2 """
3 Created on Sat Jun 29 14:38:53 2019
4
5 @author: maric
6 """
7 import wx
8 import pickle
9 import sys
10 from datetime import datetime
11 from datetime import date
12 import os
13 import copy
14
15 def ds_introduction():
16     global ev_yesno_1, ev_yesno_2, wx_text, videoplayer
17     global ev_timing, overwrit_vs_edit, file_exists, eventdata, eventstart, eventend
18     global event, event0, event1, event2, event3, event4, event5, event6, event7, event8, event9, event10, general_add_on, overwrit_vs_edit
19     global ev_1, ev_2, ev_3, ev_4, ev_5, ev_6, ev_7, ev_8, ev_9, ev_10, postict, event0, event1, event2,
20     global event5, event6, event7, event8, event9, event10, general_add_on, overwrit_vs_edit
21     global software_user_dialogue, software_user_dialogue_1, eeg, ds_starter
22     ev_1="",
23     ev_2="",
24     ev_3="",
25     ev_4="",
26     ev_5="",
27     ev_6="",
28     ev_7="",
29     ev_8="",
30     ev_9="",
31     ev_10="",
32     ev_11="",
33     postict="",
34     event0=0
35     event1=0
36     event2=0
37     event3=0
38     event4=0
39     event5=0
40     event6=0
41

```

Usage  
Here you can get help of any object by pressing **Ctrl+I** in front of it, either on the Editor or the Console.  
Help can also be shown automatically after writing a left parenthesis next to an object. You can activate this behavior in Preferences > Help.

New to Spyder? Read our [tutorial](#)

Variable explorer File explorer Help

Python console

In [1]: runfile('C:/Users/maric/.spyder-py3/Digital\_Semiology\_I4.01.2021.py', wdir='C:/Users/maric/.spyder-py3')  
Software:  
Today's date: 2021-01-14  
23:46:38  
Sergej  
Dear Colleague!  
Welcome to Digital Semiology!  
Please write your name in the row above and then press this button

At this step, both playing\_a\_video and Digital Semiology codes are running in the two separate terminals.

### *Setting the ictal episode code*

Now user can open DS and proceed to the step of ictal episode code. In order to direct video player to correct video-file, the ictal episode code should be name of the video-file without the filename extension (see paragraph Ictal Episode Code in the chapter INTRODUCTION).

### *Video player use confirmation*

When user runs DS and approaches step of the decision to use video-player or not, selecting option "Yes" will open video player and selecting option "No" will close the terminal of video player (see the paragraph Video Player: Yes or No in the chapter INTRODUCTION).

So, opening video player depends on following conditions:

1. Video player code should be in the same directory as DS code.
2. Correct path to video-file should be set in appropriate line in the video player code
3. The video player code should start run in the separate terminal.
4. The ictal episode code should be the name of video file (without extension).
5. When DS approaches the step of Video Player: Yes or No, user should select option "Yes".

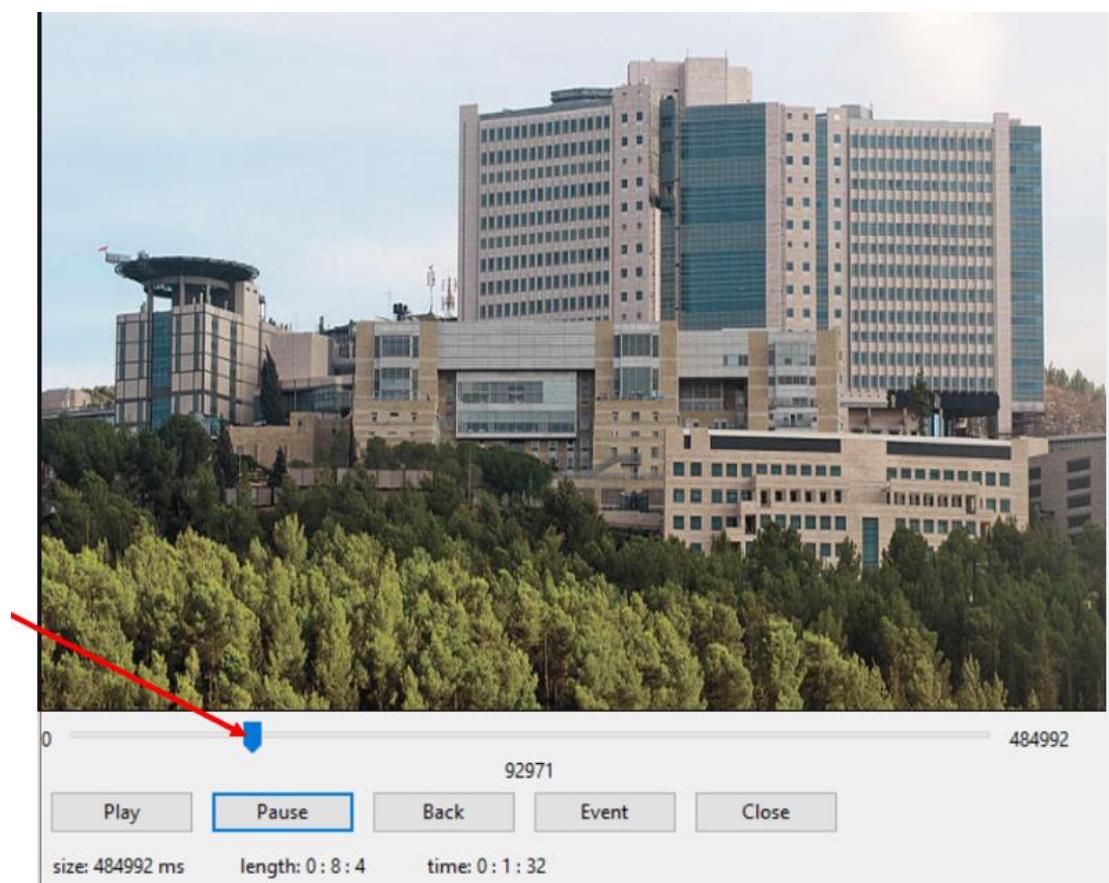
## Using Video Player

To run video, user should press Play button, to stop video – Pause button. Pause button also serves for timing indication. Another way to indicate timing is to press Event button – in that case, video will not stop.

Pressing Back button returns video to beginning.

Pressing close button will close the video player and its terminal.

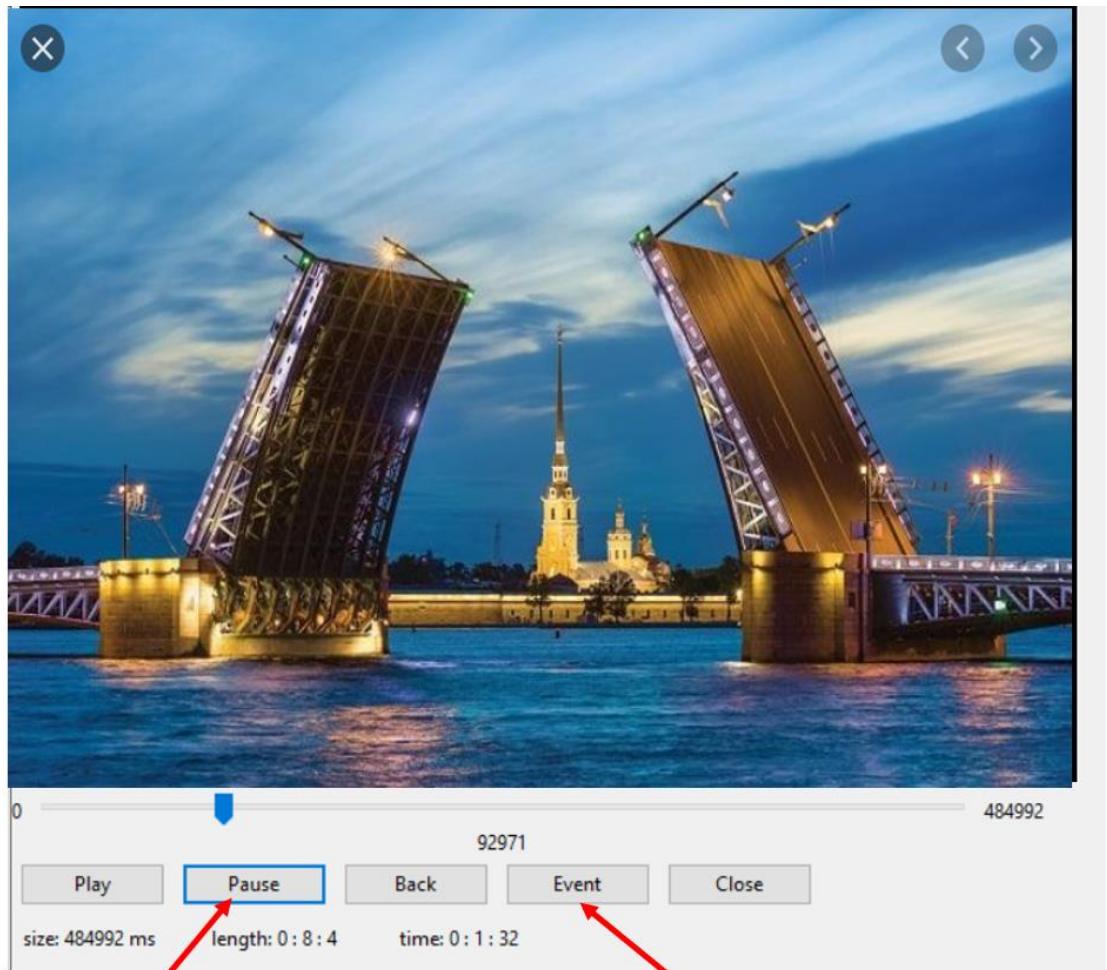
The running bar (labeled by red arrow) is moving when video runs. This running bar can be moved also manually, **the user, however, is asked to press Pause button before manually moving running bar. If running bar will be moved manually while video is running it will temporarily prevent video player functioning.**



The timescale of video player is started from 0. *Size* is number of milliseconds in the video (the same value is also displayed at the right end of running bar row). *Length* is hours : minutes : seconds in the video. *Time* is the timing of the running bar in hours : minutes : seconds (the number above the interval between Back and Event buttons is *time* in milliseconds).

## Timing Indication

In order to indicate timing, the user should press Pause button or Event button. If user presses Pause, the video will stop, if – Event – will not stop.



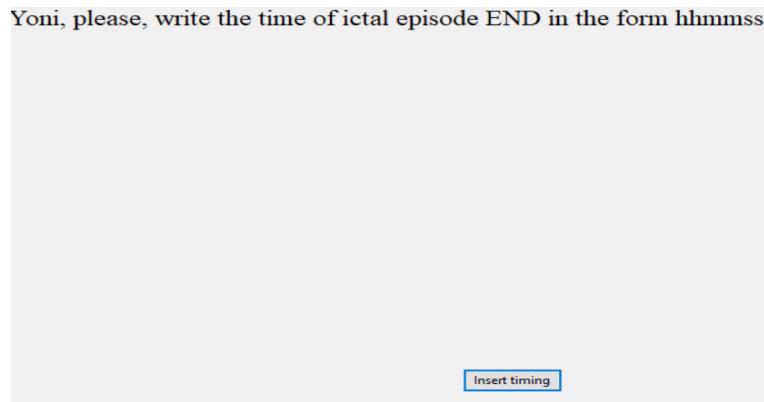
The timing indication by video player should be done when DS presents GUI asking to do that. In most cases DS asks to indicate timing of start or end of event or ictal episode. The exception is when DS asks to indicate the timing of generalized tonic clonic seizure transformation from tonic phase to clonic (see details in the paragraph Generalized Tonic Clonic Seizures in the chapter EVENTS).

Here is the GUI asking user to indicate the timing of ictal episode start. User should first press Pause or Event on video player and then press Insert timing button on DS GUI.

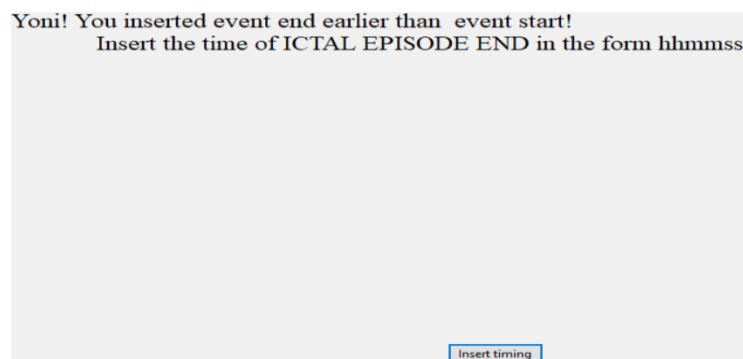
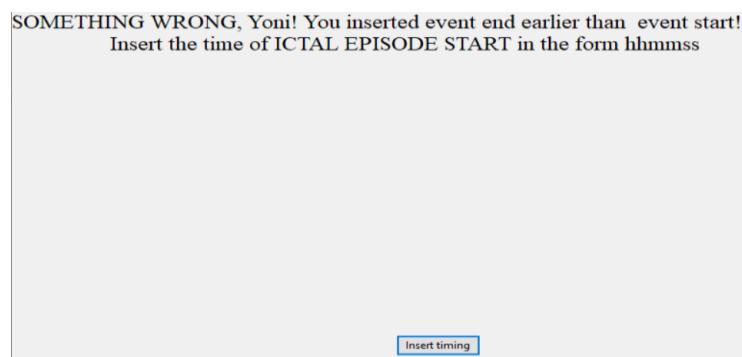
Yoni, please, write the time of ictal episode START in the form hhmmss

Insert timing

Here is the GUI asking to indicate ictal episode end. The principle is the same as for ictal episode start.



If ictal episode (or event) start is indicated later than end, DS will again ask user to re-indicate both start and end.

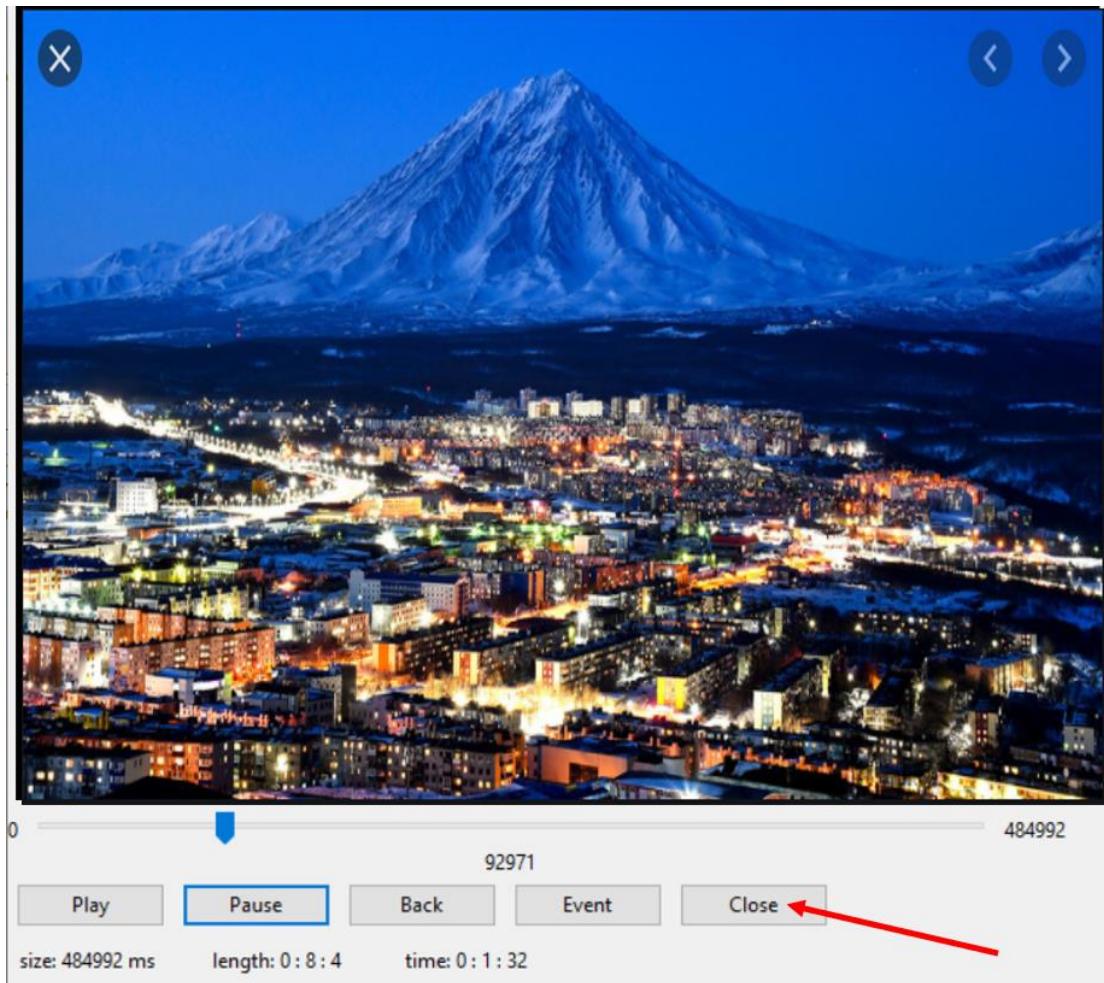


So, timing indication using video player has two requirements:

1. The timing should be indicated by pressing Pause or Event in video player at appropriate time and after that...
2. In the DS GUI the button Insert timing should be pressed.

### [Closing Video Player](#)

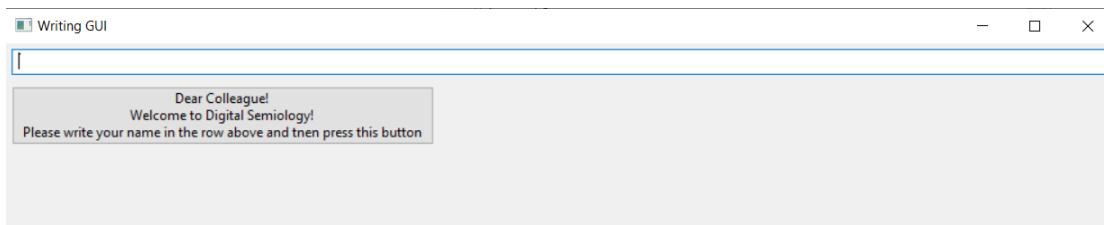
In order to close the video player, user should press the Close button – this will close both the video player and its terminal.



## INTRODUCTION PART OF DS

### Name of the User

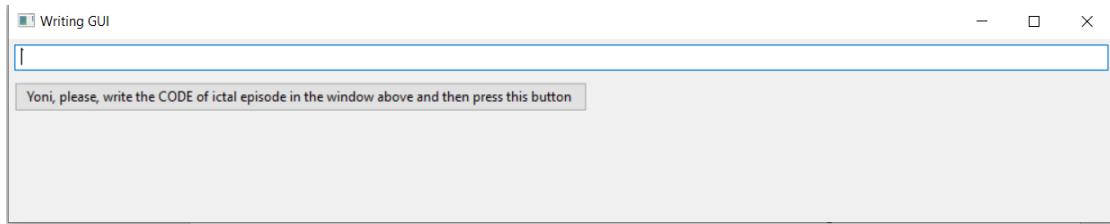
With opening DS, software asks the user to write her/his name in a writing GUI.



Letters, numbers, command punctuation characters (such as commas) can be used, but not empty spaces. Specifying user name is mandatory and progression to the next step will only be made after the user enters her/his name. When the name is written and writing GUI button is pressed, DS follows to the next step: writing the code of ictal episode.

## Ictal Episode Code

At the next step, DS asks the user to write ictal episode code in another writing GUI.



Also, here, in principle, every letter(s), every number(s) or every sign will be accepted, except for empty spaces.

When using the video-player, the ictal episode code should be the same as the name of the video-file (without filename extension). For example, if the name of video-file is 1212.wmv, the ictal episode code should be 1212 and the user should write 1212 in the row of writing GUI and then to press button.

When a .dat binary file with the same name exists in the same directory with DS code, DS will proceed to the next step, which is Overwrite or Edit.

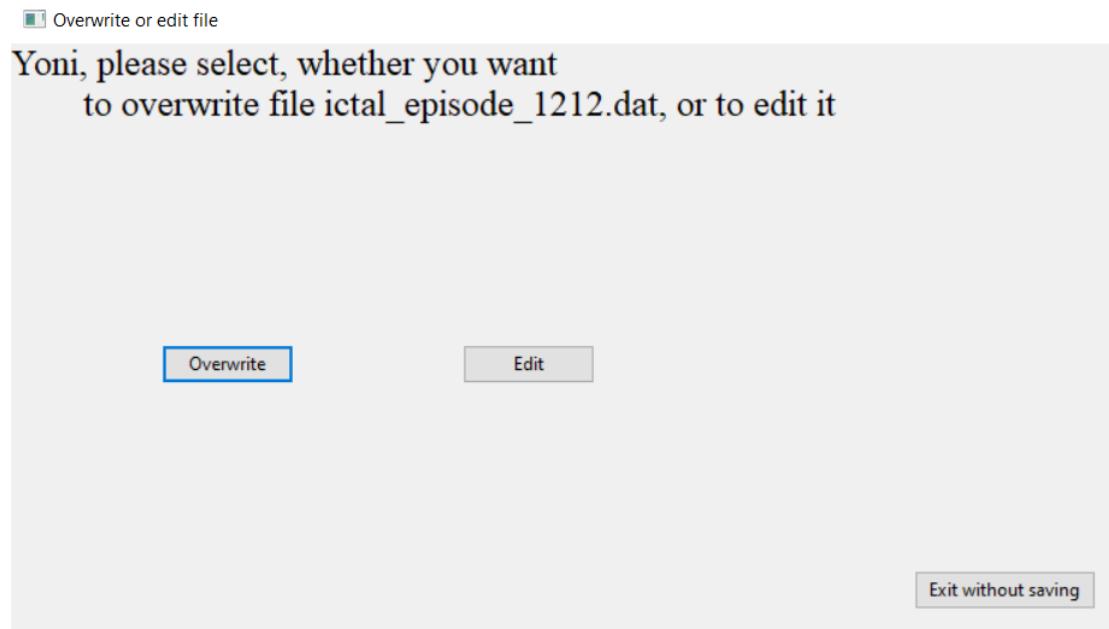
If the .dat binary file with the same name does not exist in the same directory with DS code, the software will write in IPython console the following:

"The file ictal\_episode\_1212.dat does not exist"

In this case DS will skip the step of Overwrite or Edit and will directly proceed to the next step: General Overview or Add-on Mode.

### Overwrite or Edit

This step opens only if the .dat binary file with the same name is present in the same directory as DS code. Otherwise, DS bypassing this step. At this step selecting GUI is opening.



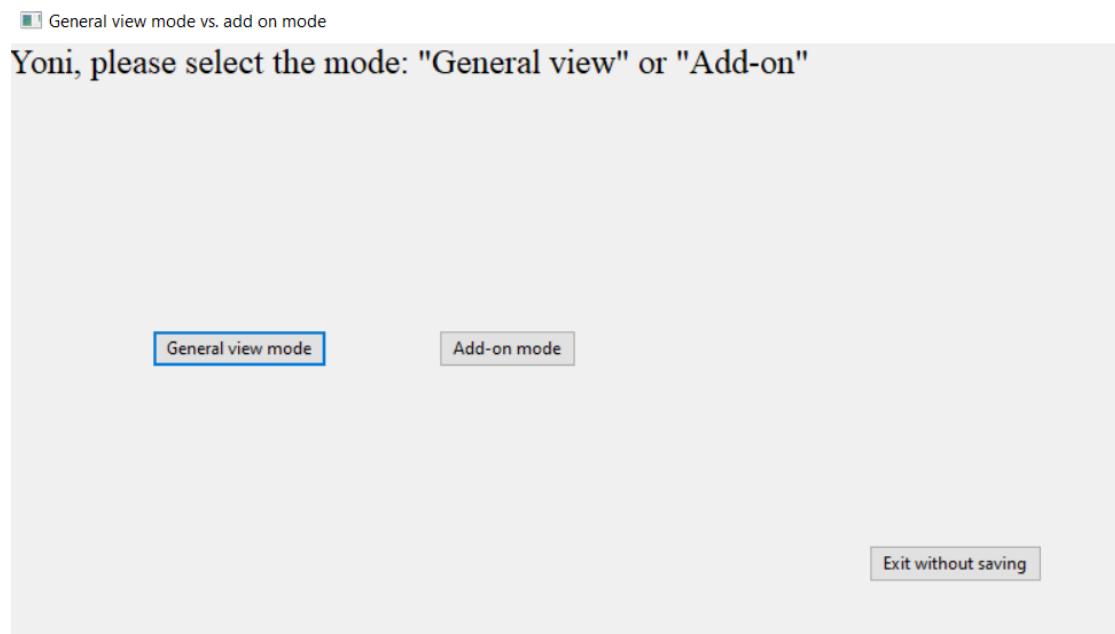
The user is asked to choose whether she/he wants to overwrite or to edit the existing .dat binary file. In the case of selection of Overwrite option, the ictus and report will be rewritten and the old information in these variables (ictus and report) will be replaced by new information. The software\_user\_dialogue variable will be not rewritten, rather updated and will contain both old and new information. If Edit option is selected, no variable in the .dat binary file will be overwrite – all three variables (ictus, report and software\_user\_dialogue) will be updated and will contain both old and new information.

If Overwrite is selected, then DS will subsequently go through all steps of Introduction, whereas, if Edit is selected, DS will bypass part of Introduction (see DS workflow).

If Overwrite option is selected, DS will continue to the next step: General View or Add-on Mode. If Edit option is selected, DS will bypass General View or Add-on Mode and will proceed directly to Machine Decisions step and the software will enter the Add-on mode.

#### [General View or Add-on Mode](#)

DS opens this step if .dat binary file (with the same name) doesn't exist or if the file exist, but the user selected Overwrite option. If file exists and user selected Edit option at previous stage, DS enters Add-on mode and doesn't provide opportunity to select General View mode. At this stage the selecting GUI appears.



The user is offered to choose one out of two operation modes: General Overview or Add-on.

The difference between these two modes is at first seen, when DS come to the step of Events and later during the Ictal Episode Editing. During Introduction there is no difference between these two modes.

General View mode allows user to describe events in two stages. At first, the user define what types of events are recorded during whole ictal episode and how many events of each type were recorded. Next, DS opens to user each event in sequence for the detailed description.

Add-on mode allows user open separate event, describe it (without listening all events), then open next one and so on, until the user describes that all events are already described.

According the authors experience with DS, in most situations, Add-on mode is more practical and is generally recommended as a main operating mode.

More details about these two modes of events description can be found in the chapter EVENTS.

After the operating mode is selected by user, DS proceeds to the next step, which is Machine Decisions: yes or no?

#### [Machine Decisions: Yes or No?](#)

This is an obligatory step which is included in all pathways in DS. Here DS presents to user the selecting GUI dealing with machine decisions.

Yes No or Exit without saving

**Yoni, if you want that Digital Semiology will  
make decisions, push "yes", if not -"no"**

Yes

No

[Exit without saving](#)

If user selects Yes at this step, DS will make and report decisions regarding focality of the ictal episode, presence of motor Jacksonian march and PNES features. If the selection is No – DS will not make such decisions. Then DS pass to next step which is Video Player: yes or no?

#### [Video Player: Yes or No?](#)

This step is also obligatory (included in all pathways of DS) as a previous one. At this step user selects whether she/he will use DS-integrated video player during ictal episode analysis. DS presents at this stage appropriate selecting GUI.

Yes No or Exit without saving

**Yoni, if you want to use the video-player,  
push "yes", if not -"no"**

Yes

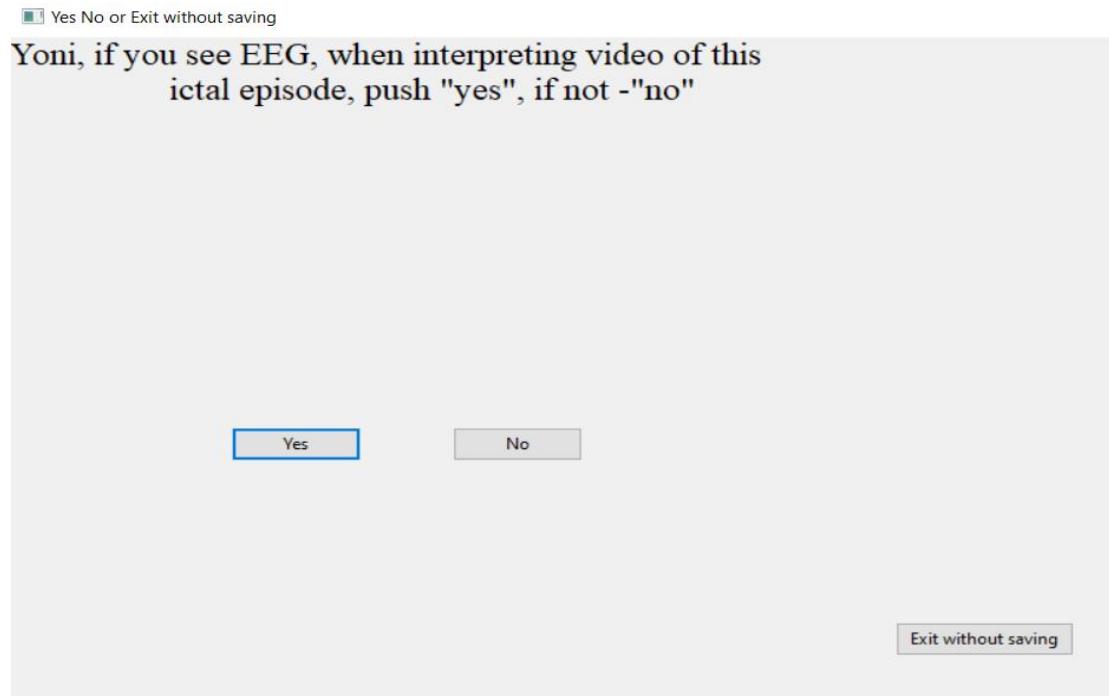
No

[Exit without saving](#)

If user selects No, video player does not open, and the user should label timings of events manually. If user selects Yes, DS opens video player, but this requires several conditions (see in chapter OPERATING VIDEO PLAYER).

#### Looking at EEG, when interpreting video

At this step the DS user is asked: whether she/he see EEG during video interpreting.

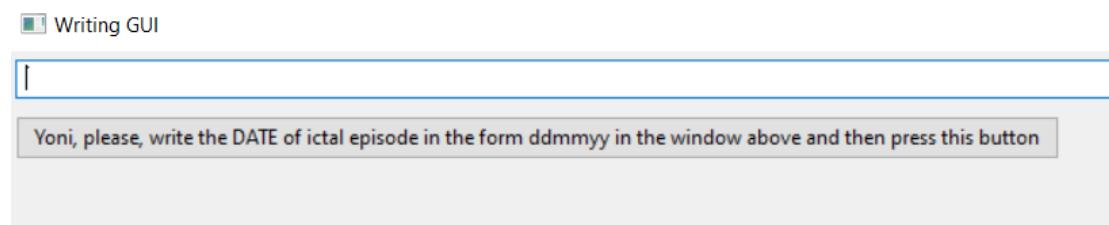


The next steps of DS depend on two conditions: 1. Whether the .dat file is new and 2. What was the user selected at one of the previous steps: Overwrite or Edit.

After the user selected Yes or No at this step, if it is new .dat binary file, or if option Overwrite (not Edit) was selected, DS continues to the next step, which is Ictal Episode Date. If it is an existing .dat binary file and option Edit was selected, DS from this point will bypass several steps of Introduction and will continue directly to Events.

#### Ictal Episode Date

At this step, DS opens writing GUI, asking user to write the date of ictal episode in the form ddmmyy.



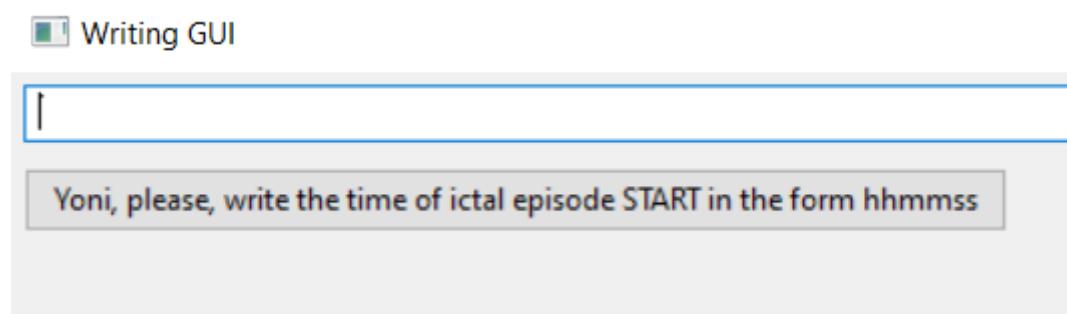
Later, in report the date will be converted to dd.mm.20yy. For example, if user write 300920, DS will generate date 30.09.2020 in the report.

If user will write the combination of numbers, which cannot be converted to the date, such as 340811 (34.08.2011) or 122520 (12.25.2020), DS will not continue to further step, rather write in IPython console the error message and will again present to user the same writing GUI until reasonable date will be written.

After this step is correctly completed, DS proceeds to next two sequential steps which are Timings of Ictal Episode Start and End.

#### [Timings of Ictal Episode Start](#)

This step is different with and without video player. When user doesn't employ video player, she/he should manually write the timing in writing GUI in the form ssmmhh. Then DS converts this to ss:mm:hh. For example, if user wrote 013305, DS will write in report 01:33:05.



Using video player, it is no need to write timing (for details see chapter OPERATING VIDEO PLAYER).

If user will write number combination, which cannot be converted to reasonable timing, error message will appear in IPython console and DS will again open the writing GUI, asking to write the reasonable timing.

After completing these steps, DS will ask user several questions regarding the general characteristics of ictal episode.

#### [Baseline Behavior?](#)

DS asks the user, whether this ictal episode can be just a baseline behavior.

DS opens a selecting GUI and gives four options: "No chance", "Possibly", "Certainly" and "No opinion".

Degree of certainty

Yoni, please define the chance that this ictal episode is just a baseline behavior.

No chance      Possibly      Certainly      No opinion

Exit

After user answered the question, DS continues to the next question: PNES?

PNES?

PNES is psychogenic non-epileptic seizure. This step is similar to previous one. Here is the GUI:

Degree of certainty

Yoni, please define the chance that this ictal episode episode is PNES.

No chance      Possibly      Certainly      No opinion

Exit

Pathologic non-psychogenic non-epileptic?

Degree of certainty

Yoni, please define the chance that this ictal episode episode  
is pathologic non-psychogenic non-epileptic.

No chance

Possibly

Certainly

No opinion

Exit

Whether ictal episode started from sleep?

Degree of certainty

Yoni, please define the chance that this ictal episode episode started from sleep.

No chance

Possibly

Certainly

No opinion

Exit

Cognitive/motor Testing During Ictal Episode?

In contrast to previous steps, the user is offered to answer this question in the form  
Yes or No (or No opinion).

Cognitive or motor testing

Yoni, if cognitive and/or motor testing was performed during this ictal episode,  
push "yes", if not -"no"

Yes

No

No opinion

Exit

After this step, DS proceeds to Events. However, here we will discuss now how to operate video player and thereafter we will continue to Events.

### Hierarchical description of ictal semiology

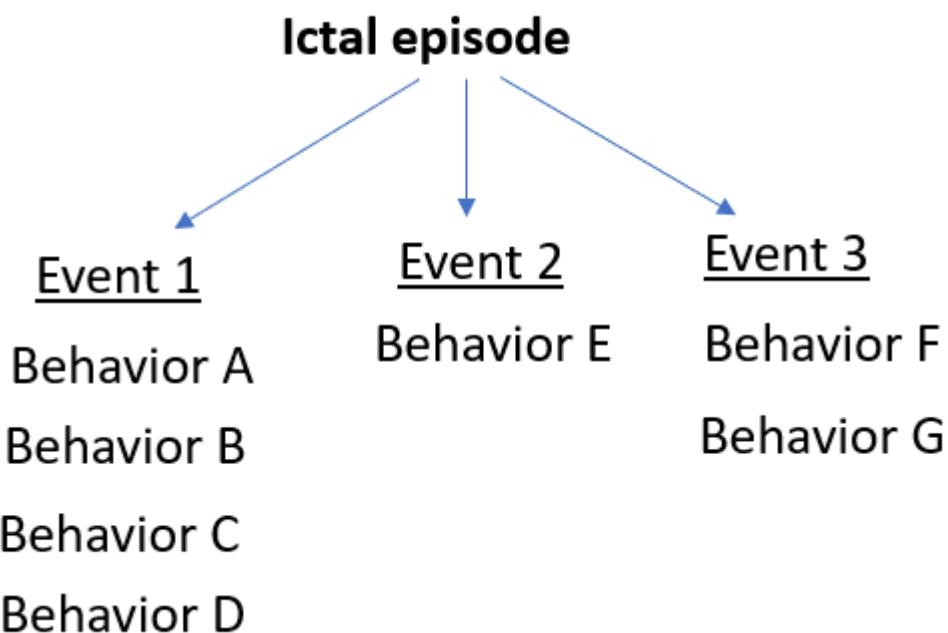
DS describes ictal episode in hierarchical way. There are 3 hierarchical categories:

1. ictal episode
2. ictal events
3. ictal behaviors

The first hierarchical category - **ictal episode** itself is characterized by the timings of its beginning and the end, separating it from baseline behaviors and postictal phenomena.

The second hierarchical category - **ictal events** are discrete behavioral elements of ictal episode. The ictal events are categorized as simple motions, automatisms, hyper-motor phenomena and others. Every ictal event has its own start and end timings.

The third (and the lowest) hierarchical category is **ictal behaviors**. Ictal behaviors are the elements of ictal events and have the same start and end timings as the ictal event. In other words, ictal behaviors are elementary semiology entities, which occur simultaneously and belong to the same ictal event type. For example, the automatic ictal event can include simultaneously occurring oral automatic behavior and hand automatic behavior.



## EVENTS

### Events: general aspects

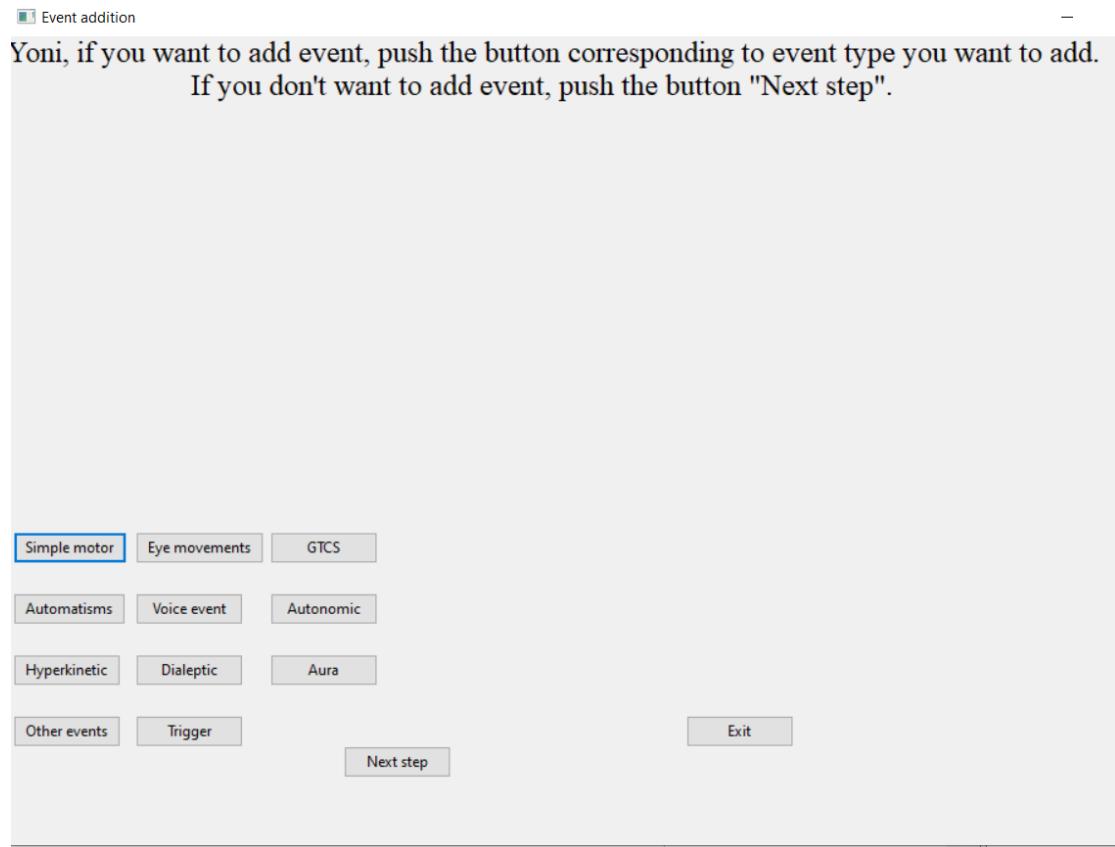
Events are elements of ictal episode, which have its own timing. DS uses following classification of events:

1. Simple motor events
2. Automatisms
3. Hypermotor events
4. Eye movement events
5. Voice events
6. Dialectic events
7. Autonomic events
8. Generalized Tonic Clonic events
9. Aura reporting events
10. Other events
11. Trigger events

In addition to its own timing, events can have their characteristics: behaviors. For example, simple motor event may include behaviors: tonic position of left arm and clonic movement of right arm. DS has different ways to describe events: one is add-on mode, and another is general view mode.

### Events: add-on mode

In add-on mode, user should choose only one event, describe it and then chose another event (in contrast to general view mode, when user choose several events at once). In add-on mode the events' GUI includes buttons, which correspond to different types of events, Next Step button and Exit button.



Pressing one of the event type buttons leads to opening GUI of timing indication (event start and after that – event end) and next – GUI of behaviors related to selected event type. After user finalized describe the event, DS returns to events' GUI.

Pressing Next Step button leads to proceeding to editing ictal episode part of DS.

Pressing Exit button initiates the process of exit from DS.

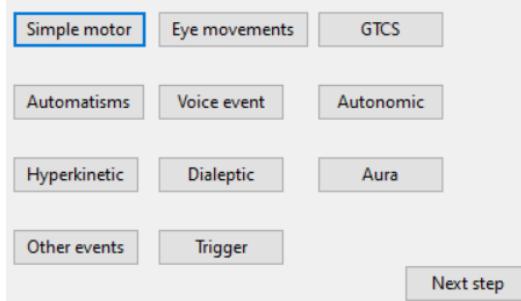
#### [Events: general view mode](#)

In general view mode, user can first select several events, then indicate how many events of a given type are included into the ictal episode, and next to describe each event separately.

The events' GUI in general view mode is similar to one in the add-on mode except of absence of Exit button.

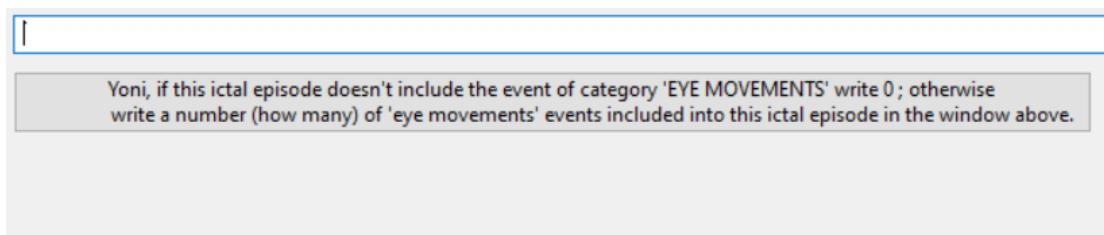
#### Types of ictal events

Yoni, push the buttons corresponding to event types included into this ictal episode, events triggering this ictal episode and (possibly) postictal events.  
You can choose more than one event type (more than one button).  
At the end push button "Next step"



Simple motor   Eye movements   GTCS  
Automatisms   Voice event   Autonomic  
Hyperkinetic   Dialectic   Aura  
Other events   Trigger   Next step

If user selects event(s) and press Next step button, DS will open writing GUI asking user to write how many events of given type are included in this ictal episode.



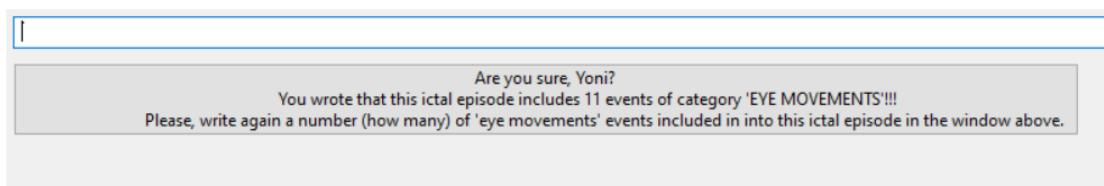
|

Yoni, if this ictal episode doesn't include the event of category 'EYE MOVEMENTS' write 0; otherwise write a number (how many) of 'eye movements' events included into this ictal episode in the window above.

DS will continue to open this GUI for all types of events that user selected.

If user writes 0, DS will not open this type of event for description.

If user writes more than 10 events, then DS will warn her/him that he indicated too many events and will ask her/him to write again the number of events of given type.



|

Are you sure, Yoni?  
You wrote that this ictal episode includes 11 events of category 'EYE MOVEMENTS'!!!  
Please, write again a number (how many) of 'eye movements' events included in into this ictal episode in the window above.

Then DS will accept any number of events that user will write.

Next DS asks the user, whether it is possible that some events are postictal.

**Yoni, if it is possible that some of the events represent postictal state,  
push "yes", if not -"no"**

**Yes**

**No**

After user selected the answer Yes or No, DS opens sequentially GUIs of all events that user previously defined.

## Abbreviations and terms used in DS

With the appearance of some Selecting GUIs, DS presents some terminology/abbreviations clarifications in IPython console.

DS uses several abbreviations:

PNES – psychogenic non-epileptic seizure (line 477).

PNPNE – pathologic non-psychogenic non-epileptic (line 503).

Polit-fist – politician's fist (line 1630 and line 1741).

Foreign language – the patient speaks, but the interpreter does not understand the language (line 5345).

PMLS – periodic limb movements of sleep (line 7755).

RLS – restless leg syndrome (line 7758).

## BEHAVIORS

Behaviors are the elements of DS report, which are next lower hierarchical category after events. While events have their time characteristics: start and end, behaviors have not their own time characteristics and accept timings of event. In other words, behaviors are characteristics of event. Behaviors can be described in five dimensions:

1. Type of behavior (e.g. tonic or clonic movements).
2. Body parts involved in a behavior (e.g. neck or right arm).

DS uses also terms *distal* or *proximal* extremity. The mining is distal arm: elbow and more distally; and distal leg: knee and more distally.

3. Power of the behavior (e.g. strong or weak)
4. Completeness of behavior, especially regarding posturing (e.g. complete or partial)
5. Details of movement or posturing (e.g. flexion [movement] or "politicians' fist" [posturing]).

Not all behaviors are described in all five dimensions.

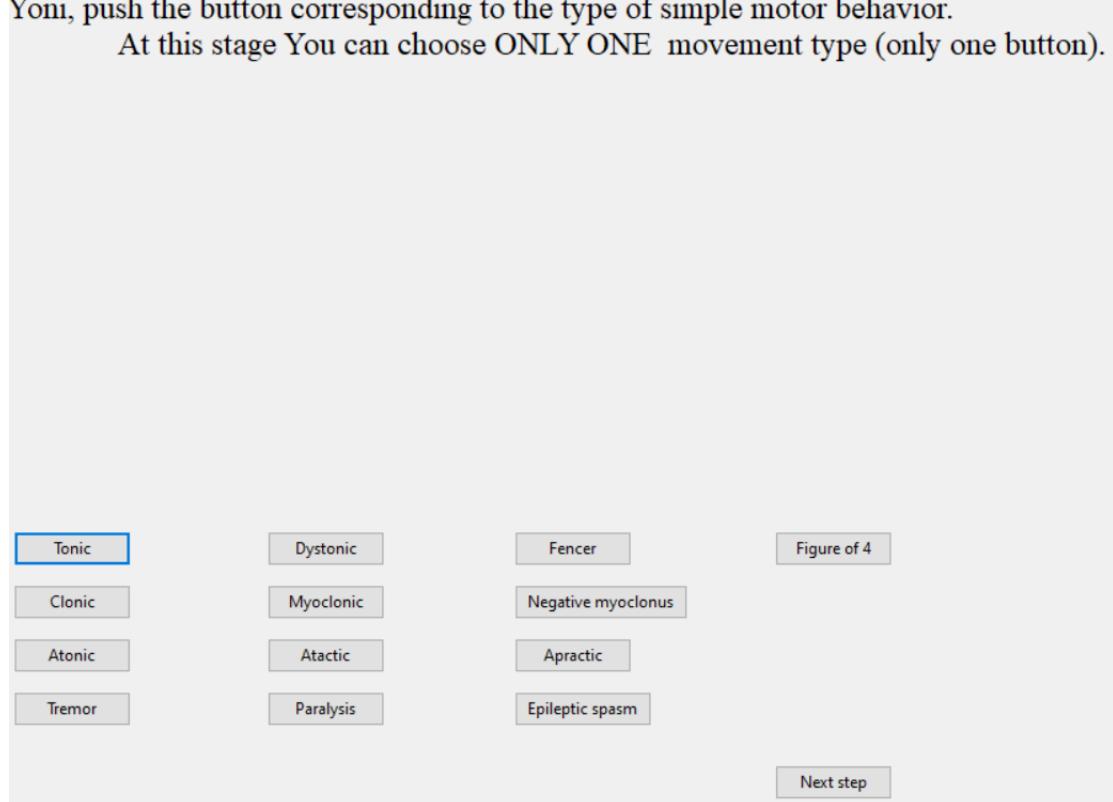
### SIMPLE MOTOR BEHAVIORS

Simple motor behaviors are motor behaviors that cannot be part of normal movement repertoire of the humane (in contrast to automatisms) – in other words they are always pathologic behaviors, which can be part of epileptic seizure (or postictal phenomena) or part of nonepileptic event.

After choosing simple motor event on the events' GUI and inserting event start and end timings, DS presets to user selecting GUI with different types of simple motor behaviors.

 Types of simple motor behaviors

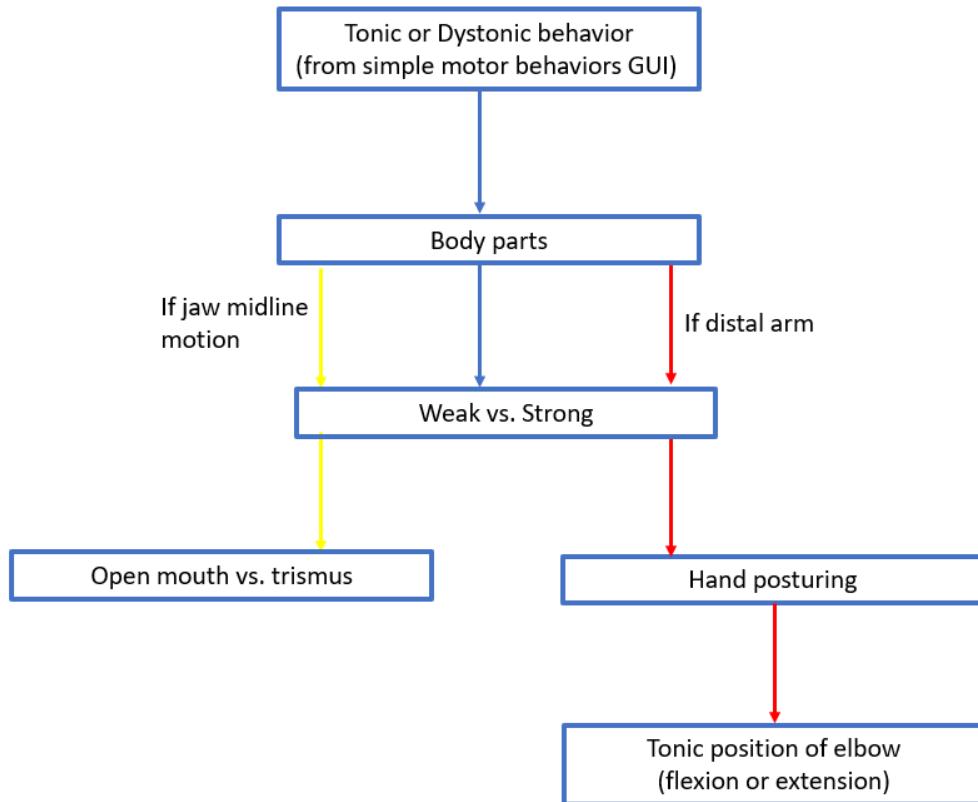
Yoni, push the button corresponding to the type of simple motor behavior.  
At this stage You can choose ONLY ONE movement type (only one button).



Tonic	Dystonic	Fencer	Figure of 4
Clonic	Myoclonic	Negative myoclonus	
Atonic	Atactic	Apractic	
Tremor	Paralysis	Epileptic spasm	

Next step

## Tonic and dystonic behaviors



These behaviors are associated with increased muscle tone. Regarding the difference between tonic and dystonic behaviors, the authors recommend choosing dystonic option, when rotatory component is present.

After choosing tonic or dystonic behavior in events GUI, DS opens a body parts GUI:

Simple motor behaviors: body parts

**Yoni, push the buttons corresponding to body parts involved in this behavior.  
You can choose more than one body part (more than one button).  
At the end push button "Next step".**

R arm prox	R leg prox	Neck to R	Trunk to R	Jaw midline	Tongue midline	Generalized
R arm dist	R leg dist	Neck to L	Trunk to L	Jaw to R	Tongue to R	Rt hemibody
L arm prox	L leg prox	Neck forw	Trunk forw	Jaw to L	Tongue to L	Lt hemibody
L arm dist	L leg dist	Neck backw	Trunk backw			
R face	L face		Next step			

At the next step, DS ask to select, whether the movement was weak or strong.

 Weak vs. strong

Yoni, please assess the power of distal part of right arm movement:  
select button "weak" or "strong".

If distal arm was selected, DS will next open the following GUI:

 Hand posturing

Yoni, push the button corresponding to the type of RIGHT hand posturing.  
You can choose ONLY ONE posturing type (only one button).

And next - the following GUI:

Tonic position of Elbow

**Yoni, push the button corresponding to the type of RIGHT ELBOW  
tonic/dystonic position: in flexion or in extention.  
You can choose ONLY ONE posturing type (only one button).**

[Elbow in flexion](#)

[Elbow in extension](#)

[Next step](#)

If user selected jaw midline movement in body parts GUI, the user is asked first to select whether the movement is weak or strong and later DS ask her/him to select what is the jaw movement: mouth opening or trismus:

Open mouth vs. trismus

Yoni, if there is mouth opening - push "Open", if there is trismus - push "Trismus",  
if neither of them - push "Next step".

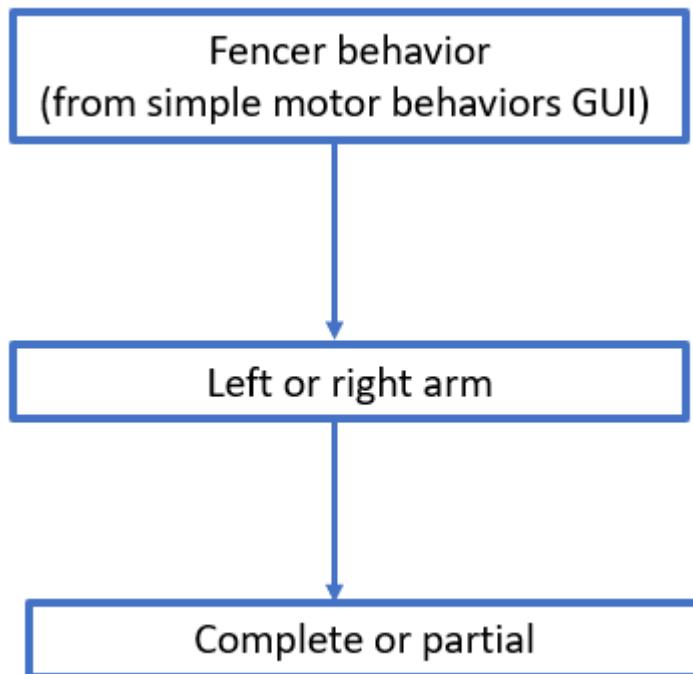
[Open mouth](#)

[Trismus](#)

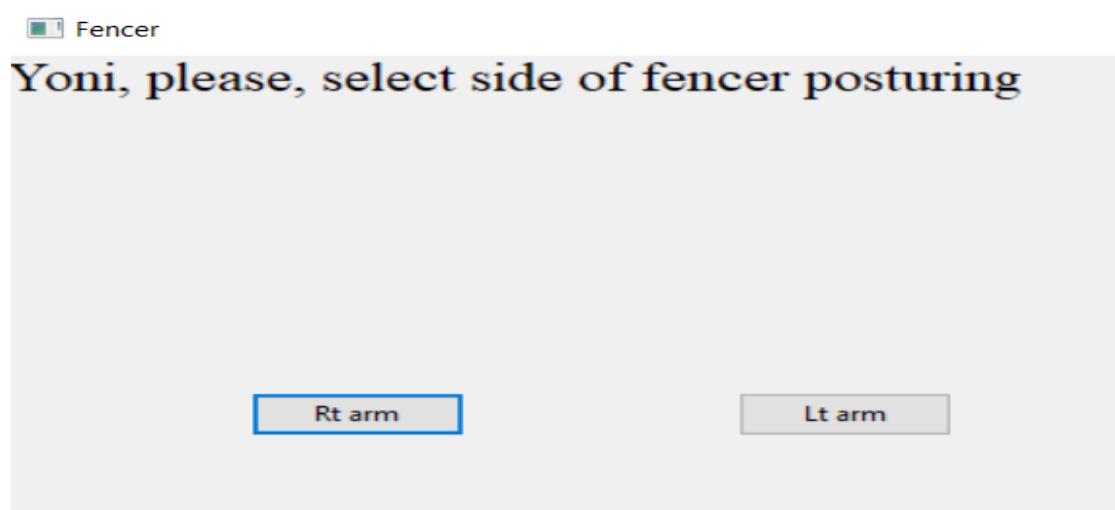
[Next step](#)

## Fencer posturing

Fencer posturing and figure-of-four posturing are both tonic movements, however, DS offers separate buttons for these entities in simple motor behaviors GUI.



If user selected fencer from simple motor behaviors GUI, DS first ask the user to select, whether fencer posturing involves right arm or left arm.



At the next step, DS asks the user, whether fencer posturing is complete or partial.

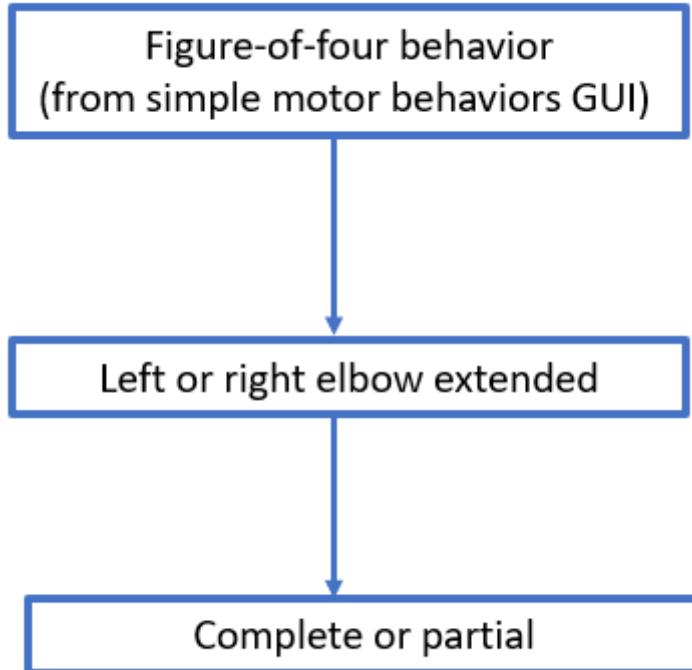
 Complete vs. Partial

Yoni, please, select whether fencer posturing is complete or partial.

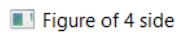
Complete

Partial

### Figure-of-four posturing



If user selected figure-of-four from simple motor behaviors GUI, DS first ask the user to select, what is the side of extended elbow.

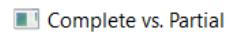


Yoni, please, select side of elbow extension in figure-of-4.

Rt elbow extended

Lt elbow extended

At the next step, DS asks the user, whether figure-of-four posturing is complete or partial.

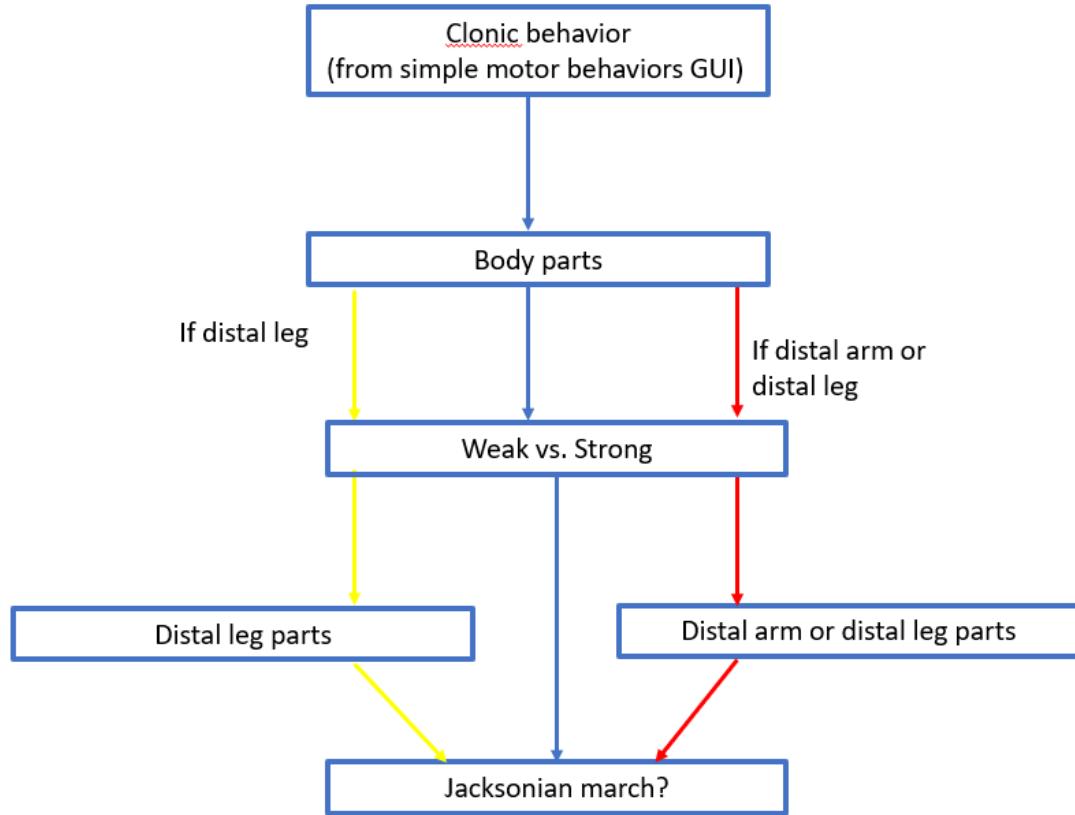


Yoni, please, select whether figure-of-4 is complete or partial.

Complete

Partial

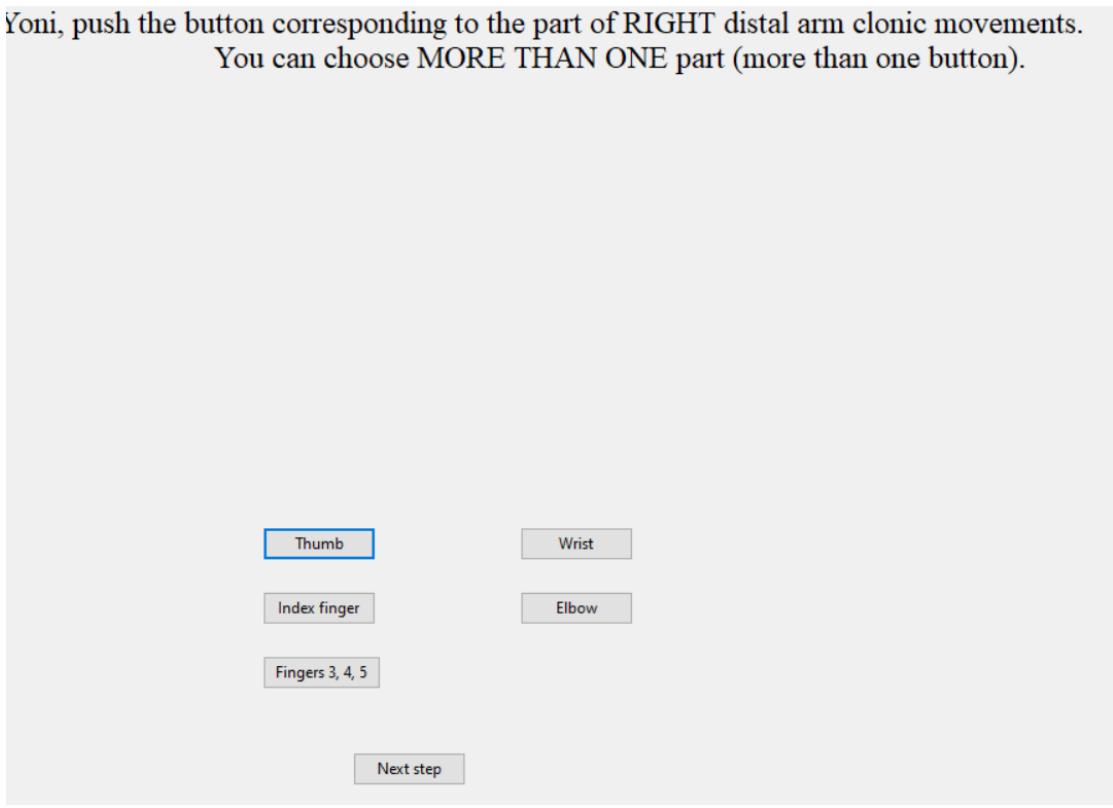
## Clonic behaviors



If user selected clonic behavior from simple motor behaviors GUI, DS presents first the body parts GUI and then Weak vs. Strong GUI. If distal arm was chosen, the following GUI will appear:

 Distal arm clonic movements

Yoni, push the button corresponding to the part of RIGHT distal arm clonic movements.  
You can choose MORE THAN ONE part (more than one button).



A user interface for selecting distal arm clonic movements. It features five rectangular buttons arranged in two rows. The top row contains 'Thumb' (highlighted with a blue border) and 'Wrist'. The bottom row contains 'Index finger', 'Elbow', and 'Fingers 3, 4, 5'. Below these buttons is a single 'Next step' button.

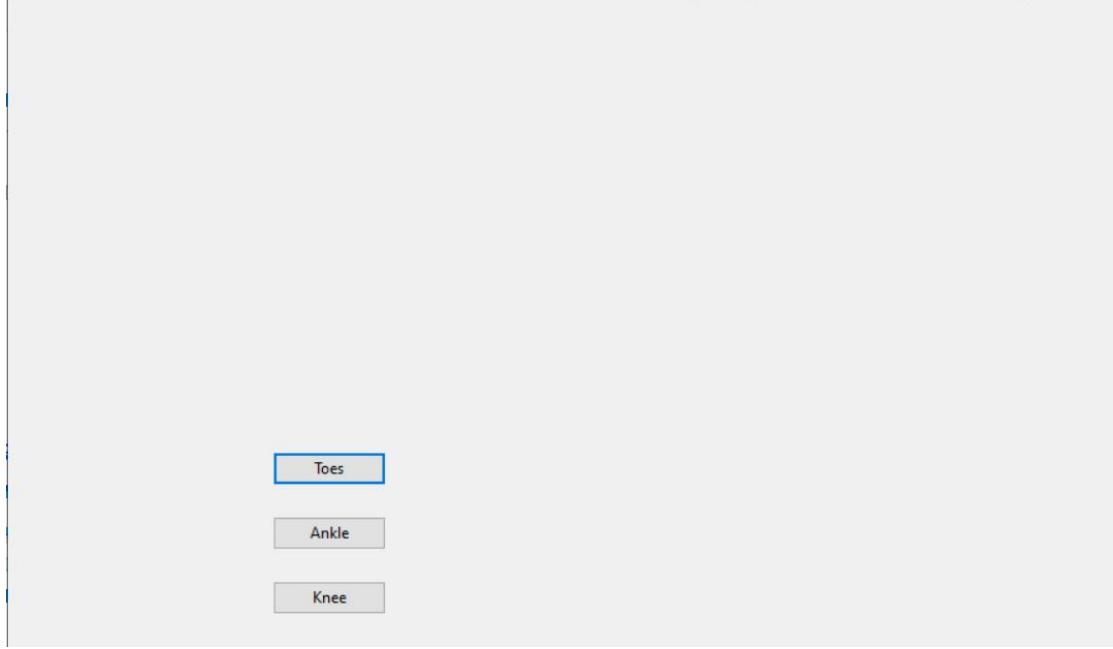
Thumb	Wrist
Index finger	Elbow
Fingers 3, 4, 5	

Next step

If distal leg was selected, then DS opens the following GUI:

 Distal leg clonic movements

Yoni, push the button corresponding to the part of RIGHT distal LEG clonic movements.  
You can choose MORE THAN ONE part (more than one button).



A user interface for selecting distal leg clonic movements. It features three rectangular buttons arranged vertically: 'Toes' (highlighted with a blue border), 'Ankle', and 'Knee'.

Toes
Ankle
Knee

Next, DS asks the user, whether or not the patient demonstrated jacksonian march

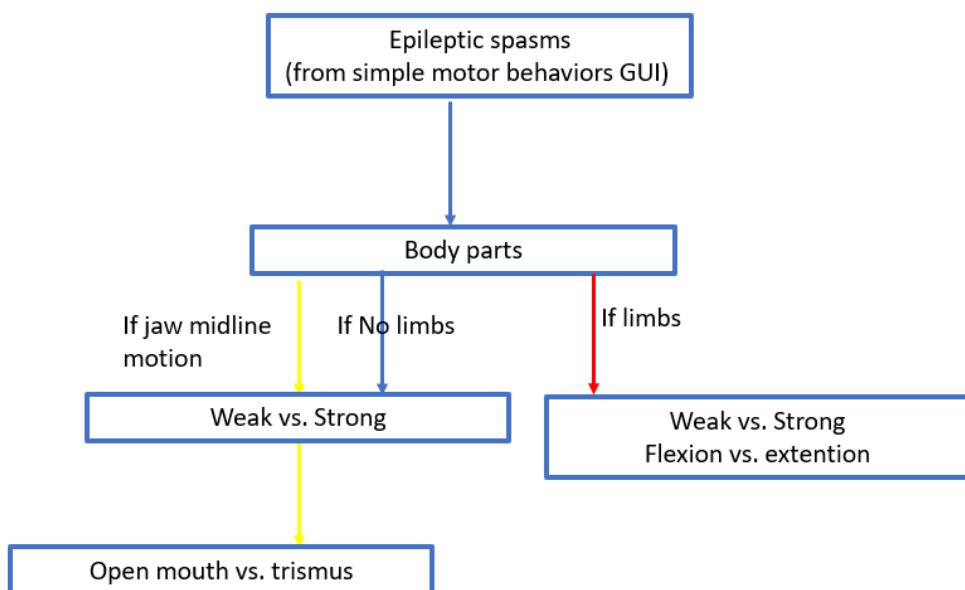
Types of simple movements

Yoni, please, select whether there is jacksonian march.

No jacksonian march      Right jacksonian march      Left jacksonian march

Next step

Epileptic spasm



If user selected epileptic spasm from simple motor behaviors GUI, DS the body parts GUI. If user selected limbs, DS asks whether it is strong or weak flexion or extension.

 Flexion vs. extension

Yoni, please assess the power of distal part of right leg movement:  
select appropriate button.

Weak flexion

Strong flexion

Weak extension

Strong extension

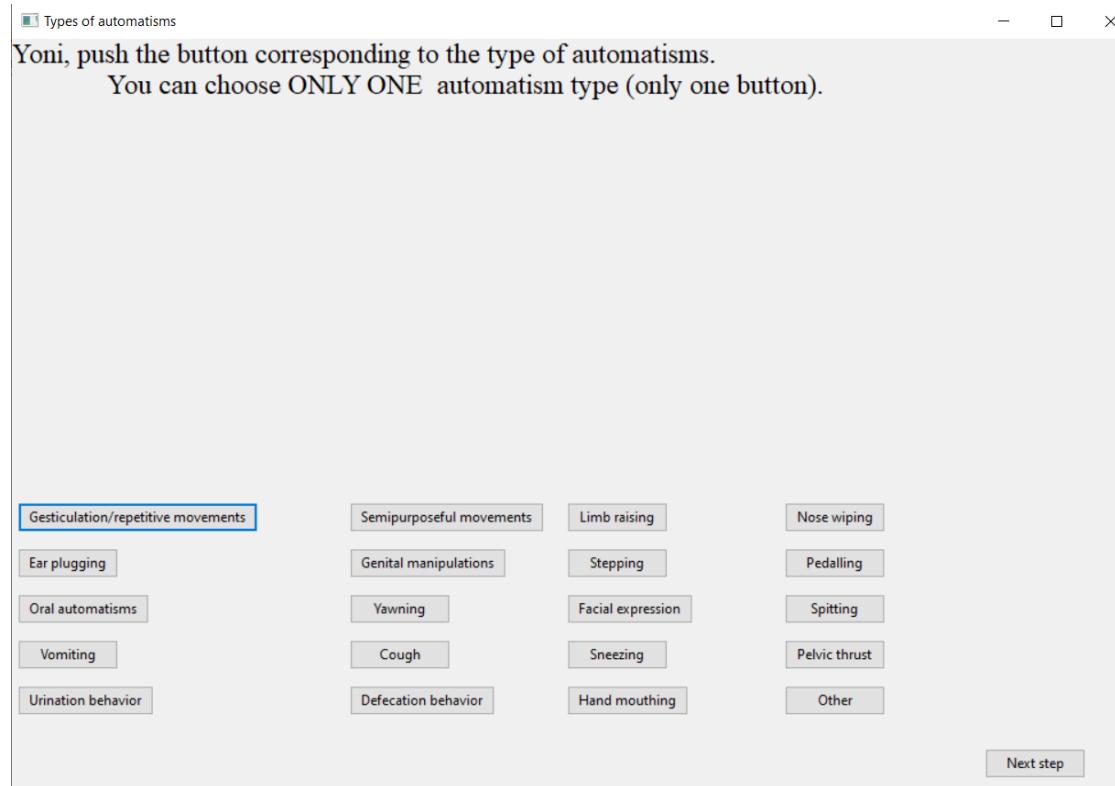
If user selected other (no limbs) body parts, DS opens Weak vs. Strong GUI.

If user selected jaw midline movement in body parts GUI, the user is asked first to select whether the movement is weak or strong and later DS ask her/him to select what is the jaw movement: mouth opening or trismus (as with tonic behaviors).

## AUTOMATISMS

Automatisms are movements, which under other circumstances can be part of normal movement repertoire. However, in the context of epileptic seizures, they are pathologic. Automatisms are often repetitive and inappropriate to the situation.

If user selected Automatism in events GUI and inserted event start and end timings, DS opens types of automatisms GUI:



It should be noted that DS allows labeling pelvic thrusting both as automatism and as "other" behavior, their it is named unspecified pelvic thrusting. The authors suggest using pelvic thrusting as automatism if user believes that it is part of epileptic seizure and as "other" behavior if she/he believes that it is PNES.

If user selected:

- Gesticulation/repetitive movements or
- Semipurposeful movements or
- Limb raising,

DS will ask user to label the limbs (right or left arm or leg) that are involved and whether the involvement is strong or weak:

 Automatisms: limbs

Yoni, push the buttons corresponding to limbs involved in this behavior.

You can choose more than one limb (more than one button).

At the end push button "Next step".

R arm strong	R arm weak
L arm strong	L arm weak
R leg strong	R leg weak
L leg strong	L leg weak
Next step	

If user selected:

- Nose wiping or
- Ear plugging or
- Genital manipulations or
- Hand mouthing,

DS will ask user to label the arms (right or left) that are involved and whether the involvement is strong or weak.

 Automatisms: arms

Yoni, push the buttons corresponding to limbs involved in this behavior.

You can choose more than one limb (more than one button).

At the end push button "Next step".

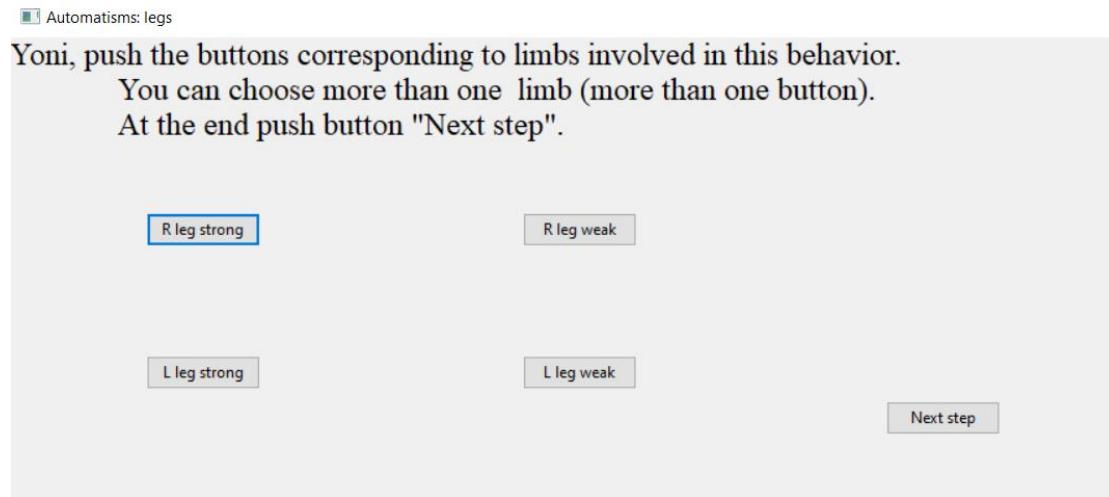
R arm strong	R arm weak
L arm strong	L arm weak
Next step	

If user selected:

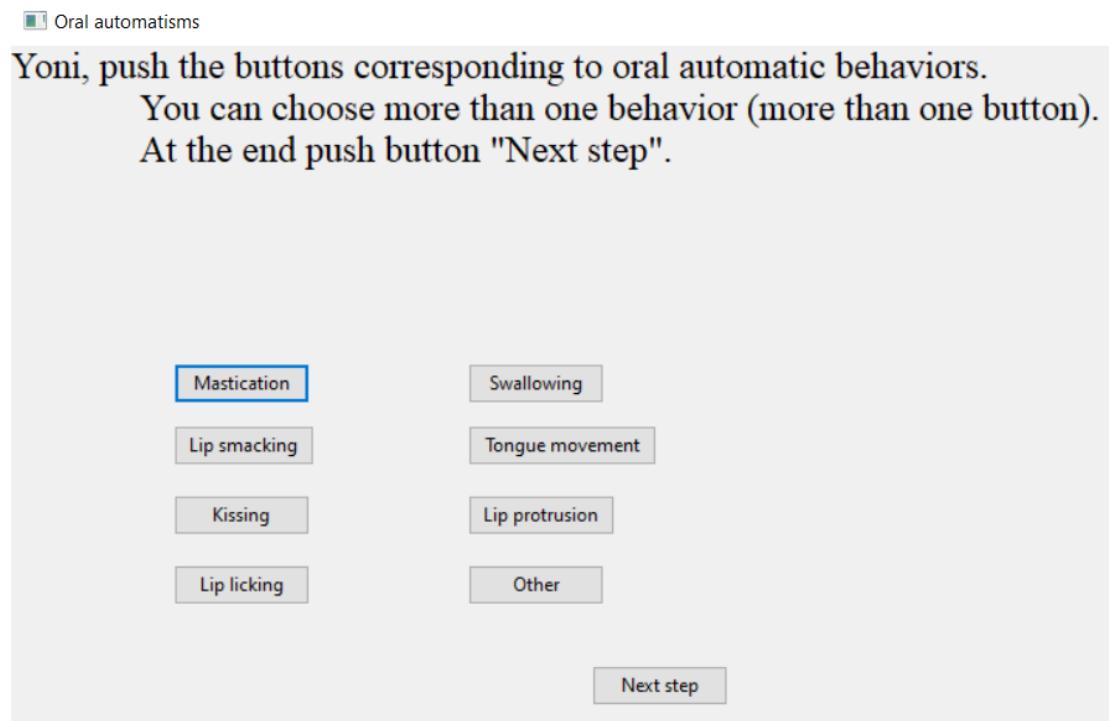
- Nose wiping or
- Ear plugging or
- Genital manipulations or

- Hand mouthing,

DS will ask user to label the legs (right or left) that are involved and whether the involvement is strong or weak:



If user selected oral automatism in types of Automatisms GUI, DS opens oral automatisms GUI:



If user selected facial expression in types of Automatisms GUI, DS opens Automatisms: facial expression GUI:

 Automatisms: facial expression

Yoni, push the button corresponding to facial expression.  
You can choose ONLY ONE behavior (only one button).

Staring	Disgusting	Happy	Sad
Smile	Shape de gendarm face	Laughter grimacing	Fear
Uncspecified facial expression		Other facial expression	Unspecific grimacing

**Next step**

## HYPERKINETIC BEHAVIORS

When user selects hyperkinetic behaviors from event GUI and inserts event start and end timings, DS opens the hyperkinetic behaviors GUI.

 Hyperkinetic behaviors

Yoni, Push the buttons corresponding to hyperkinetic behaviors.  
You can choose more than one behavior (more than one button).  
At the end push button "Next step".

Bimanual-bipedal automatism	Attempt of patient to prevent movements
Gyratory to the right	Gyratory to the left
Rocking back and forth	Rocking side-to-side

**Next step**

Next, DS asks user, what are the body parts, which are involved especially prominently in hyperkinetic behavior.

Other or hyperkinetic event: body parts

Yoni, push the buttons corresponding to body parts  
that are especially prominently involved in this behavior.  
You can choose more than one body part (more than one button).

R arm

L arm

R leg

L leg

Face

Neck/Head

Trunk

Next step

#### GENERALIZED TONIC CLONIC SEIZURES

If user selected GTCS from event GUI and inserted event start and end timings, DS will ask her/him to inset the time of transition from tonic phase to clonic. GTCS is only event type, which requires inserting three timings and not two (event start and event end).

Next DS will ask user to select the body position at the end of seizure:

GTCS end/turn position

**Yoni, push the button corresponding to position at the end of GTCS.  
You can choose only one position (only one button).**

Next, DS asks to select the body position after turn the patient after the end of seizure. If patient did not turn after the seizure, the authors suggest selecting the same body position as in the previous GUI.

GTCS end/turn position

**Yoni, push the button corresponding to position after the turn after the end of GTCS.  
You can choose only one position (only one button).**

These two last questions are related to risk of SUDEP.

## AUTONOMIC BEHAVIORS

If user selected "autonomic" from event GUI and inserted event start and end timings, DS will open the types of autonomic phenomena GUI:

Types of autonomic phenomena

Yoni, push the button corresponding to the types of autonomic phenomena.  
You can choose ONLY ONE autonomic phenomenon type (only one button).

Hyperventilation	Hypoventilation	Apnea	Dyspnea
Stridor	Piloerection	Palor	Flashimg
Cyanosis	Sweating	Mydriasis	Miosis
Cheyne-Stoke respiration	Irregular respiration	Hiccups	Salivation
Urine loss*	Fecal loss*	Other	<input type="button" value="Next step"/>

It is important to note that the meaning of urine loss and fecal loss is different than urinary behavior or defecation behavior in automatisms GUI. The authors suggest considering urine loss or fecal loss as autonomic phenomena, when they are not associated with urination or defecation behavior, otherwise they should be considered as automatisms.

If user selected mydriasis or miosis, DS will open the following GUI:

 Pupil size change

Yoni, push the buttons corresponding to pupil size changes.  
 You can choose more than one behavior (more than one button).  
 At the end push button "Next step".

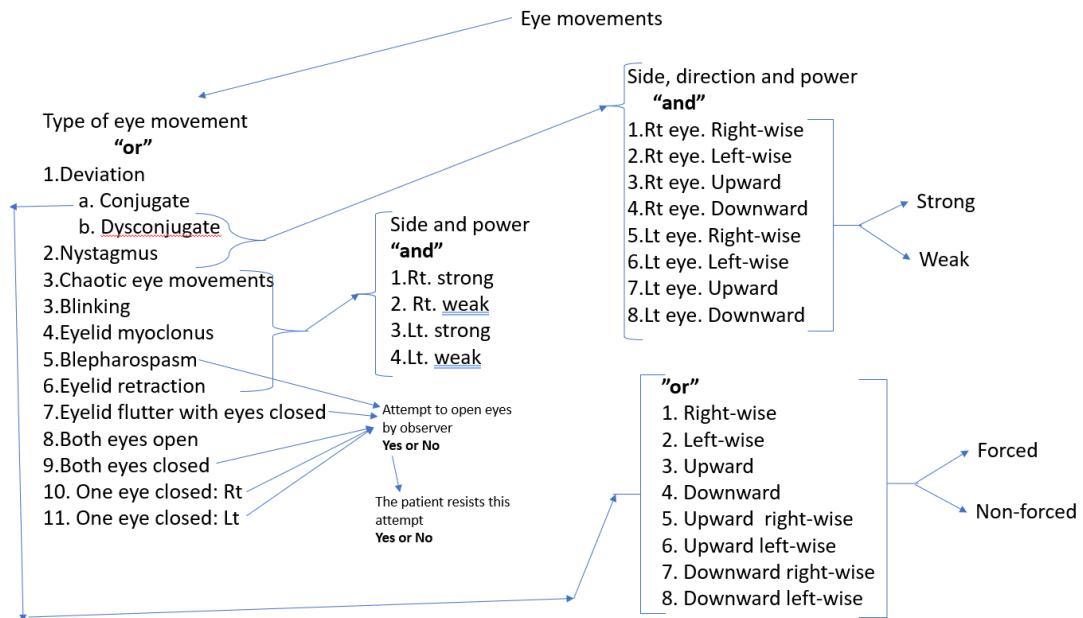
Right strong	Left strong
Right weak	Left weak
Next step	

If user selects the combination of Right strong and Right weak or Left strong and Left weak, DS will not accept such combination and will return this GUI to the user, while in the IPython console appears the following statement:

**Software:**

**YOU CAN'T PRESS BOTH 'STRONG' AND 'WEAK' OF THE SAME PUPIL!  
 PLEASE TRY AGAIN!**

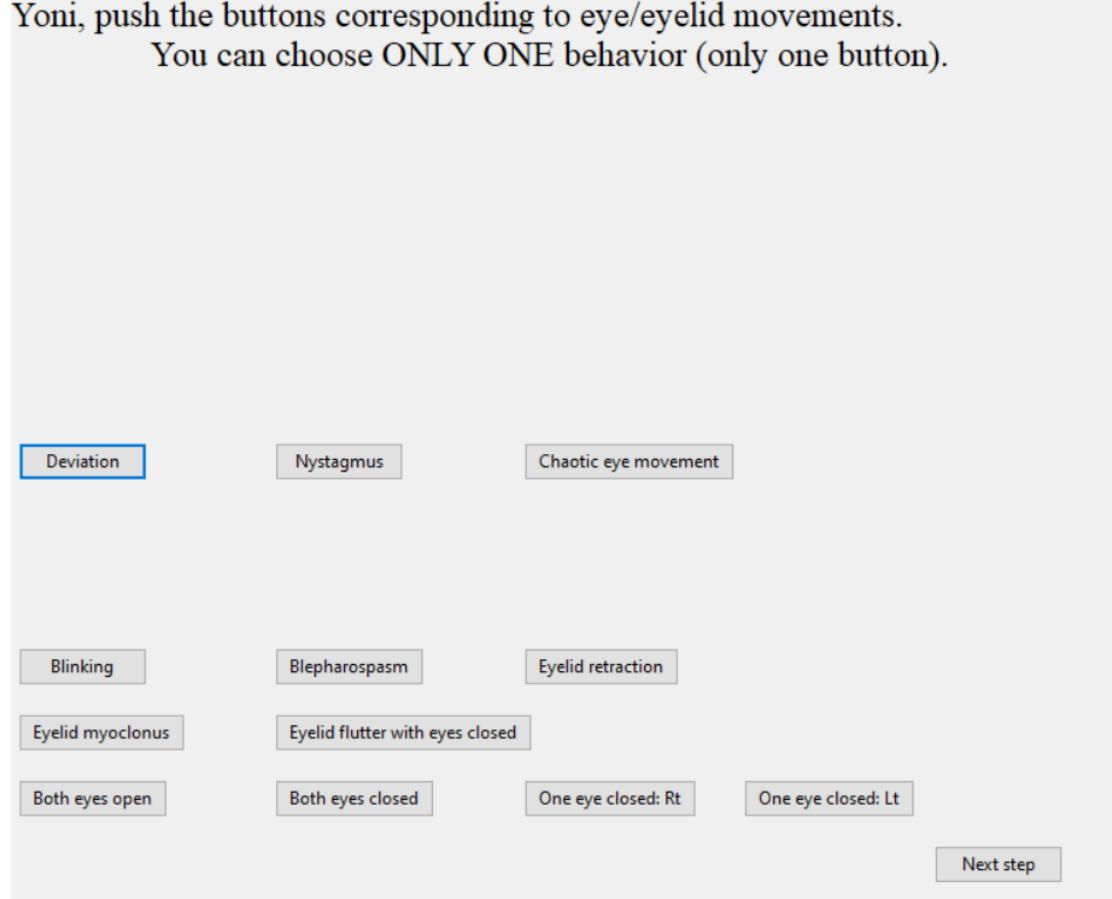
## EYE/EYELID MOVEMENT BEHAVIORS



After user selected eye movements from the event GUI and inserted event start and end timings, DS opens Eye/eyelid movement GUI. In this GUI the upper row of buttons corresponds to eye movements and the lower three rows – to eyelid movements.

 Eye/eyelid movement

Yoni, push the buttons corresponding to eye/eyelid movements.  
You can choose ONLY ONE behavior (only one button).



The GUI for eye/eyelid movement selection consists of a grid of buttons. At the top, there are three main categories: 'Deviation' (highlighted with a blue border), 'Nystagmus', and 'Chaotic eye movement'. Below these are several sub-categories and specific behaviors:

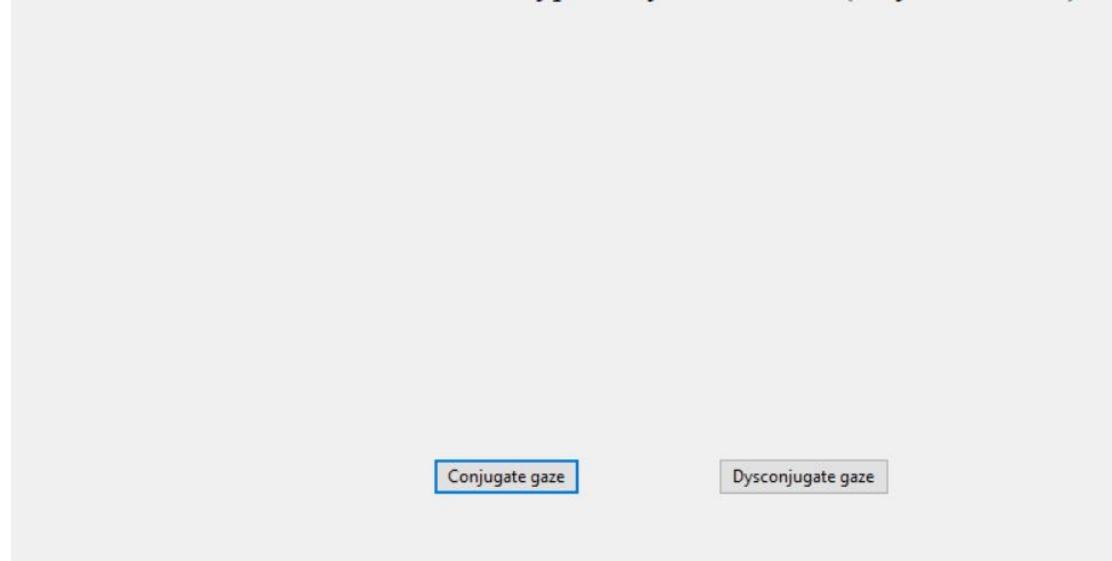
- Row 1: 'Blinking', 'Blepharospasm', 'Eyelid retraction'
- Row 2: 'Eyelid myoclonus', 'Eyelid flutter with eyes closed'
- Row 3: 'Both eyes open', 'Both eyes closed', 'One eye closed: Rt', 'One eye closed: Lt'

A 'Next step' button is located at the bottom right of the grid.

If user selects Deviation, DS presents the following GUI:

 Conjugate vs. dysconjugate eye deviation

Yoni, Select type of eye movements: conjugate  
vs. dysconjugate eye deviation  
You can choose ONLY ONE type of eye movement (only one button).



The GUI for selecting conjugate vs. dysconjugate eye deviation consists of two buttons at the bottom:

- 'Conjugate gaze' (highlighted with a blue border)
- 'Dysconjugate gaze'

If user selects conjugate gaze, DS opens the following GUI:

 Conjugate gaze deviation

Yoni, push the buttons corresponding to conjugate gaze deviation.  
You can choose ONLY ONE gaze deviation (only one button).

Up forced	Up-right forced	Up non-forced	Up-right non-forced
Down forced	Up-left forced	Down non-forced	Up-left non-forced
Right forced	Down-right forced	Right non-forced	Down-right non-forced
Left forced	Down-left forced	Left non-forced	Down-left non-forced

If user selects nystagmus or dysconjugated gaze deviation, DS opens the Side direction and power of eye movements GUI:

 Side direction and power of eye movements

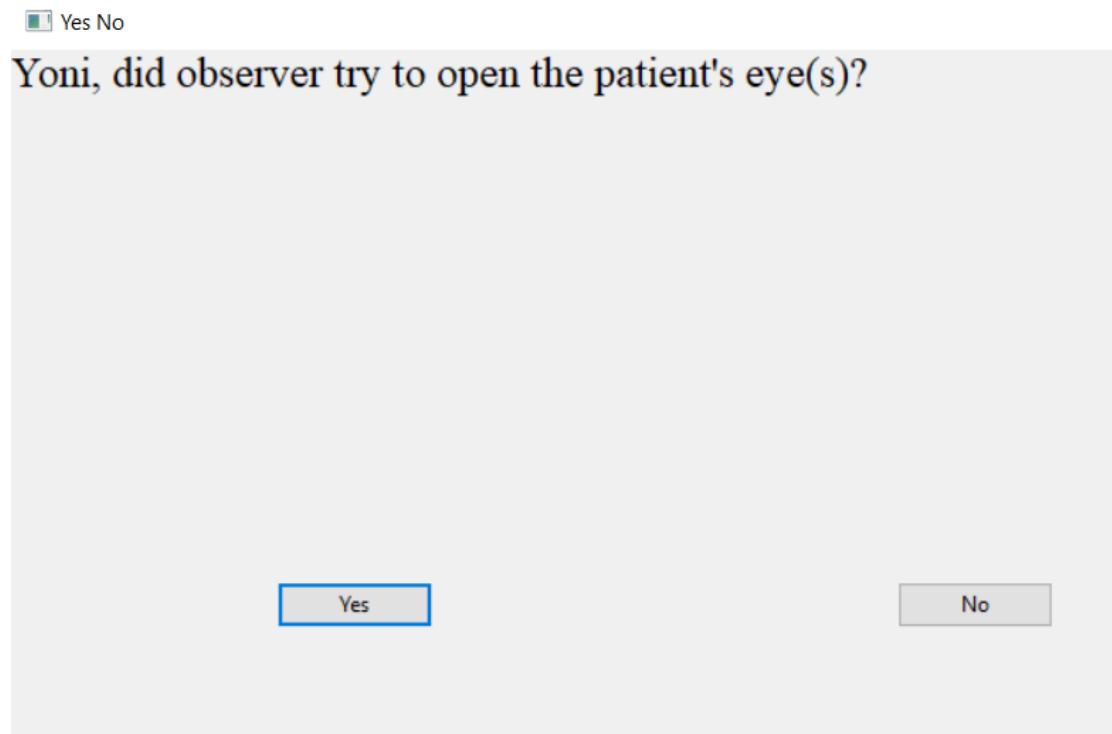
Yoni, push the buttons corresponding to side, direction and power of eye movements.  
You can choose more than one behavior (more than one button).  
At the end push button "Next step".

R eye rightwise strong	R eye leftwise strong	R eye upwise strong	R eye downwise strong
R eye rightwise weak	R eye leftwise weak	R eye upwise weak	R eye downwise weak
L eye rightwise strong	L eye leftwise strong	L eye upwise strong	L eye downwise strong
L eye rightwise weak	L eye leftwise weak	L eye upwise weak	L eye downwise weak

If user selects impossible eye movements combination (the movement of the same eye in opposite direction or both weak and strong movement of the same eye in the same direction), DS returns this GUI and the following statement appears in IPython console:

**Software:**  
**YOU SELECTED IMPOSSIBLE COMBINATION!**  
**PLEASE TRY AGAIN!**

If user selected blepharospasm, eyelid flutter with eyes closed, both eyes closed, one eye closed: Rt or one eye closed: Lt, DS presents the following GUI:



In this GUI, if user selects Yes, DS will open the following GUI:

Yes No

Yoni, did patient resist eye opening by observer?

Yes

No

If user selects chaotic eye movements, blinking, eyelid myoclonus, blepharospasm or eyelid retraction, DS opens Side of eye/eyelid movement GUI:

Side of eye/eyelid movement

Yoni, push the buttons corresponding to side and power of the eye/eyelid movements.

You can choose more than one option (more than one button).

At the end push button "Next step".

Rt strong

Lt strong

Rt weak

Lt weak

Next step

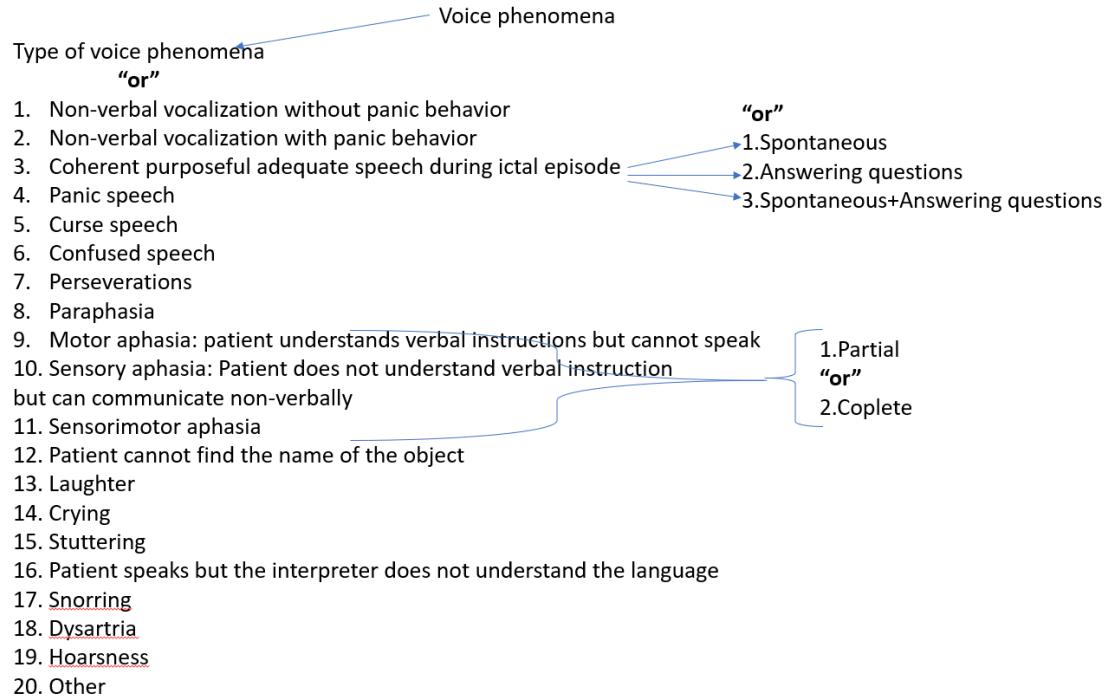
If user will select both strong and weak of the same side, DS will not accept it, will return the GUI and will write in IPython console:

Software:

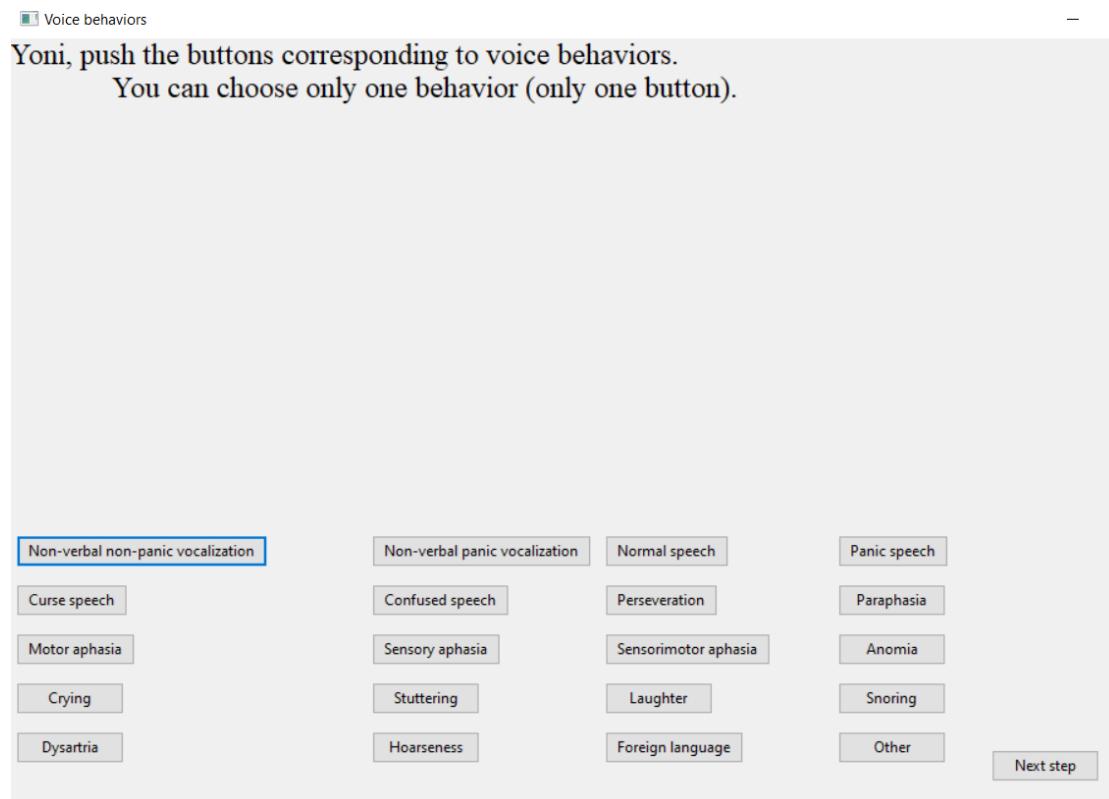
YOU CAN'T PRESS BOTH 'STRONG' AND 'WEAK' OF THE SAME EYE/EYELID!

PLEASE TRY AGAIN

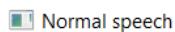
## VOICE BEHAVIORS



After user selected "voice event" from the event GUI and inserted event start and end timings, DS opens voice behaviors GUI:



If user selects "Normal speech", DS will ask her/him whether it is spontaneous speech, answering questions or both:



Yoni, please select: spontaneous speech, answering questions or both.

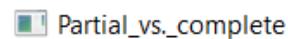
Spontaneous speech

Answering questions

Both

In the report DS will describe normal speech as: "coherent purposeful adequate speech".

If user selected "Motor aphasia", "Sensory aphasia" or "Sensorimotor aphasia", DS will ask her/him, whether aphasia is partial or complete:



Yoni, please select: partial or complete aphasia.

Partial

Complete

If the user selects the button "Foreign language" from the voice behaviors GUI, DS will include into report following statement:

"the patient speaks, but the interpreter does not understand the language".

#### DIALEPTIC BEHAVIORS

After user selected eye movements from the event GUI and inserted event start and end timings, DS opens dialeptic behaviors GUI:

#### Dialectic behaviors

Yoni, push the buttons corresponding to dialectic behaviors.

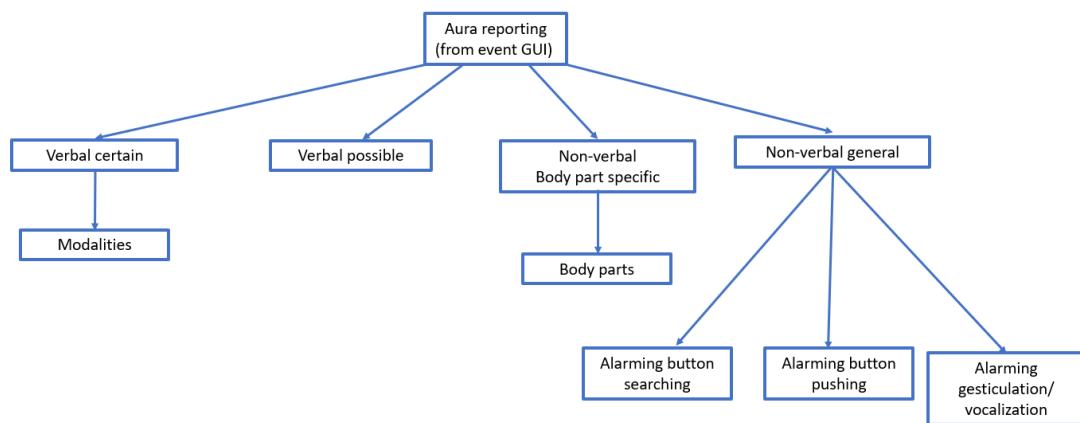
You can choose only one type of behavior (only one button).

Complete unresponsiveness	Partial unresponsiveness	Amnesia
Cessation of activity	Slowing of activity	Muscle hypotonia
Exploratory behavior	Non-verbal aggressive behavior	Non-verbal panic behavior
Agitation	Disoriented behavior	Other

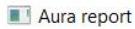
[Next step](#)

## AURA REPORTING BEHAVIORS

While aura can provide important clinical information, it is a filling, and therefore cannot be seen by itself on video, the patient however can report about aura during video/audio recording, therefore aura reporting is included in DS.



After user selected aura from event GUI, DS opens the Aura report GUI:



Yoni, push the button corresponding to aura report.

You can choose only one type of aura report (only one button).

Aura verbal report (certain)

Aura verbal report (possible)

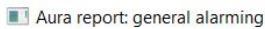
Aura non-verbal report: body part specific

Aura non-verbal report: general

Next step

When the patient clearly reports the aura by comprehensible words, the user is offered to select "Aura verbal report (certain)". If the patient tries to report aura, but it is not clear what patient says before seizure start, the user is offered to select "Aura verbal report (possible)". If the patient tries non-verbally report aura and indicates some body part, here the user is expected to select "Aura non-verbal report: body part specific". If patient report aura non-verbally by changing behavior without indicating body part, the user is offered to select "Aura non-verbal report: general".

If user selects "Aura non-verbal report: general", DS opens Aura report: general alarming GUI:



Yoni, push the button corresponding to aura report: general alarming.

You can choose only one type of aura report (only one button).

Alarming button searching

Alarming button pushing

Alarming gesticulation/vocalisation

Next step

#### Body parts in aura reporting behaviors

In relation to aura reporting behavior DS refers to body parts in two situations:

1. Aura non-verbal report: body part specific
2. Aura verbal report (certain) – somatosensory aura.

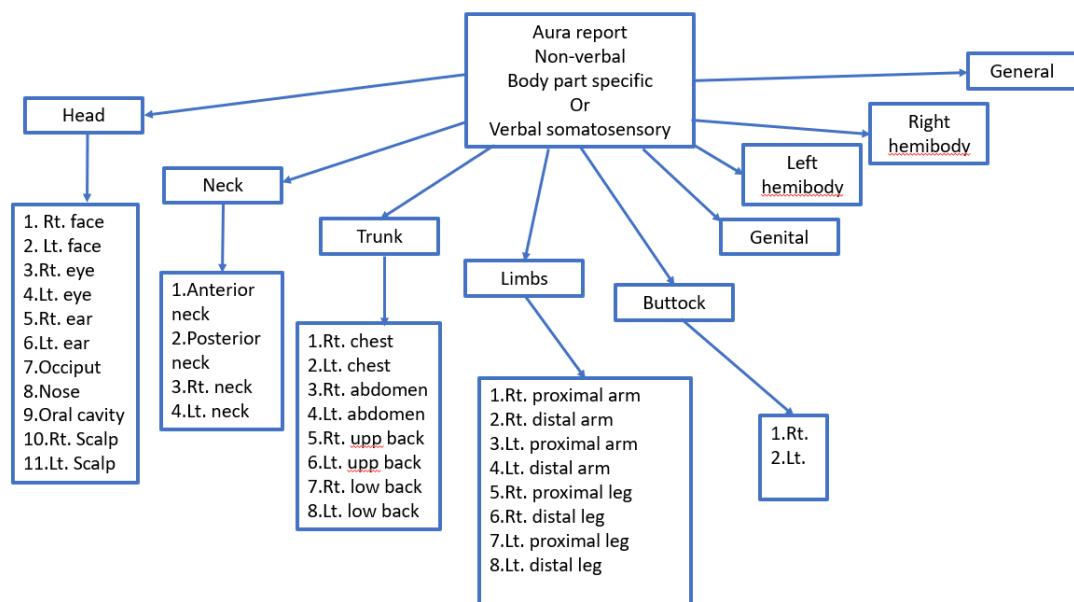
If user selects one of these two options, DS opens the following GUI:

 Aura report: body parts

Yoni, push the buttons corresponding to body parts indicated by the patient.  
You can choose more than one body part (more than one button).  
At the end push button "Next step".

Head	Limbs	Generalized
Neck	Buttock	Rt hemibody
Trunk	Genital	Lt hemibody
Next step		

Here is the schematic view of the body parts in aura reporting behaviors:



Aura verbal report (certain)

If user selects "Aura verbal report (certain)", DS opens following GUI:

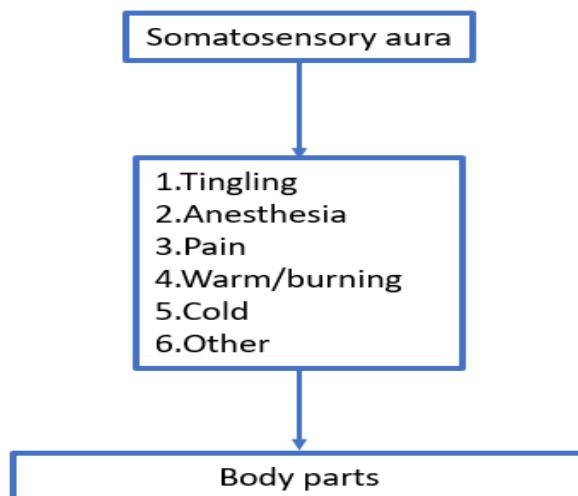
 Auras verbal report: modalities

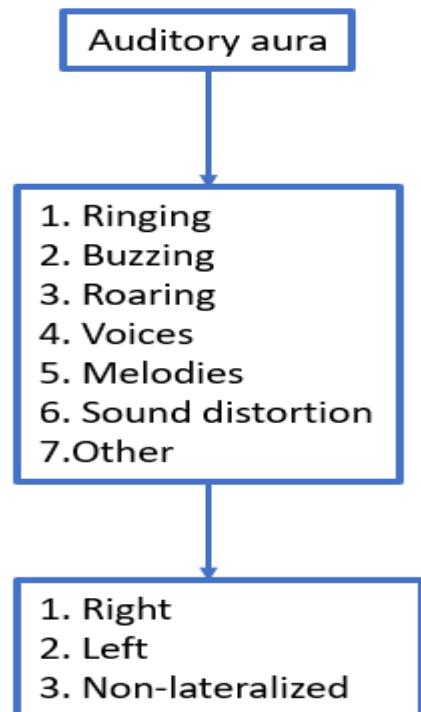
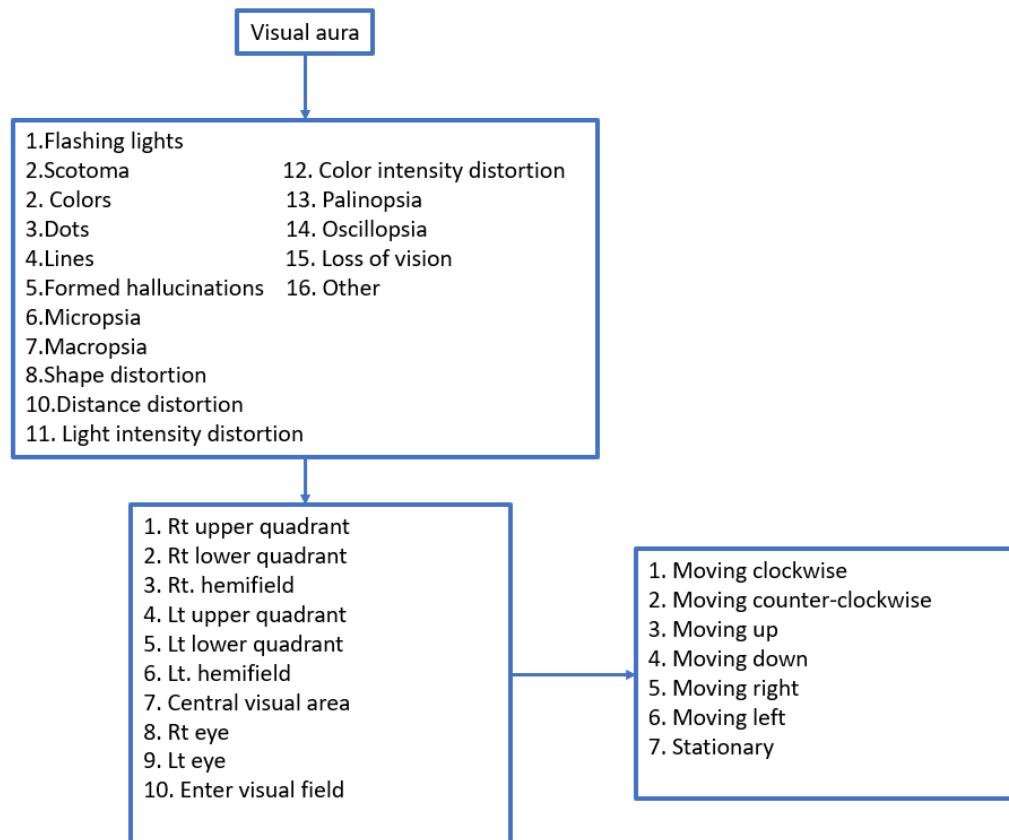
Yoni, push the buttons corresponding to type of aura reported by patient.  
At this stage, You can choose only one type of aura (only one button).

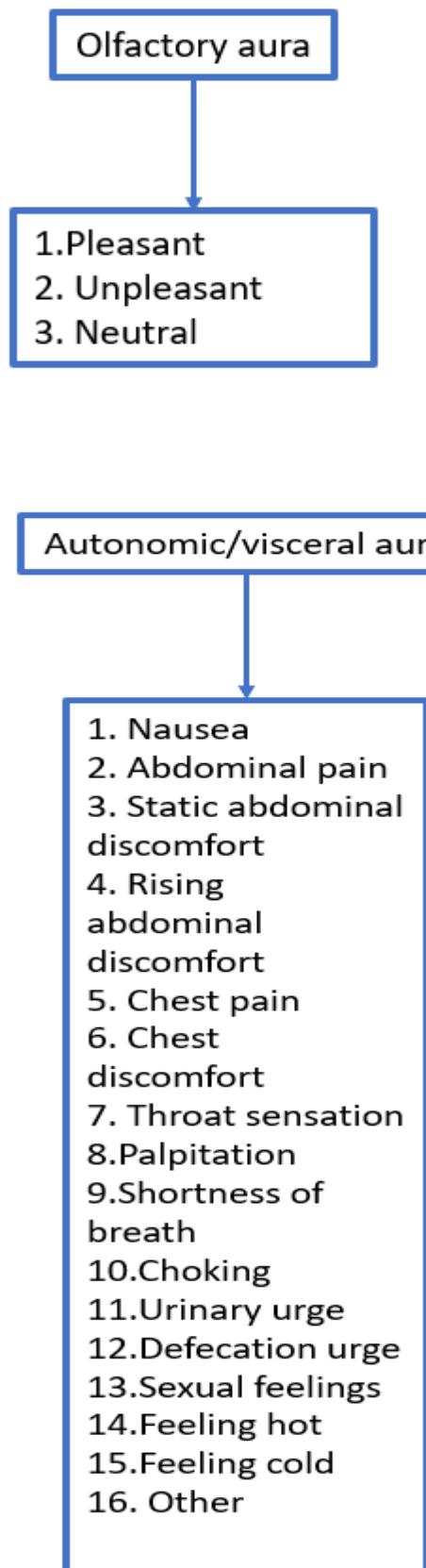
Somatosensory aura      Gustatory aura      Autonomic/Visceral aura      Cephalic aura  
Visual aura      Olfactory aura      Other aura  
Auditory aura      Vertiginous aura      Expiriential aura

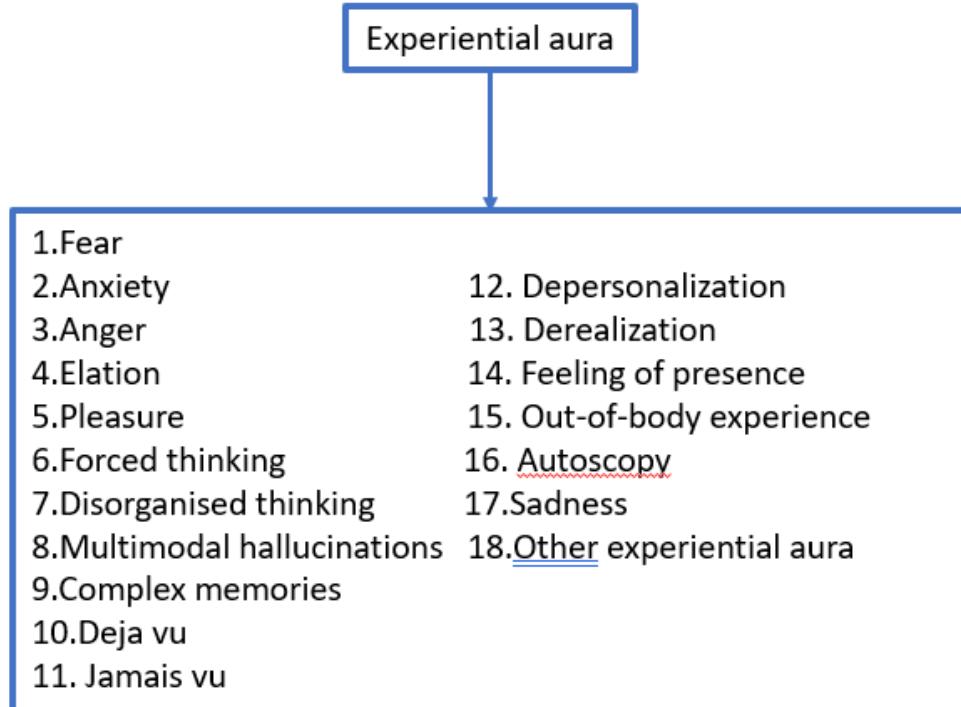
**Next step**

Here are the schematic presentations of aura modalities in DS:



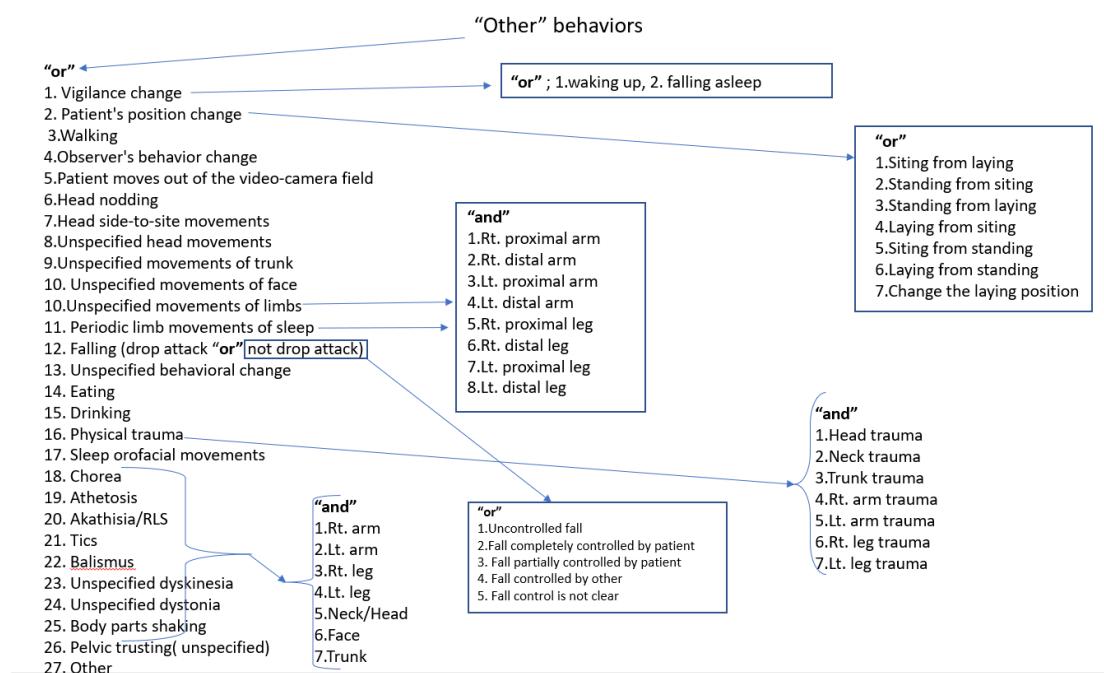






## OTHER BEHAVIORS

"Other" behaviors are behaviors that are non-epileptic or non-specific or unclassified:



If user selects "Other" from the event GUI, DS opens the "Other" behaviors GUI:

"Other" behaviors

Yoni, push the button corresponding to "other" behaviors.  
You can choose only one type of "other" behavior (only one button).

Vigilance change	Position change	Walking	Eating	Drinking
Fall	Physical trauma	Unspecified behavior change	Out of camera	Observer behavior change
Head motion unspecified	Trunk motion unspecified	Limbs motion unspecified	Face motion unspecified	Pelvic trusting (unspecified)
Head nodding	Head side-to-side motion	Body parts shaking	Electrode self-detaching	
PLMS	Sleep orofacial movements	Akathisia/RLS	Unspecified dyskinesia	Unspecified dystonia
Chorea	Athetosis	Balismus	Tics	Other

## TRIGGER

If user selects trigger from event GUI, DS opens the following GUI:

Trigger

Yoni, push button corresponding to type of trigger event.  
You can choose more than one trigger type (more than one button).

Visual	Auditory music	Auditory-no-music
Tactile	Startle	
Eating	Drinking	Other

## SEMILOGIC EXTRAPOLATION FUNCTIONS ("MACHINE DECISIONS")

DS analyses the data inserted by user and make decisions regarding four aspects:

1. Focality – whether the seizure has focal onset.
2. Motor jacksonian march
3. PNES
4. SUDEP risk

The semiologic exploration functions suggest the possibility that the features given entity (for example PNES) exist or they declare avoidance of decision. They do not suggest the absence of the given entity. In the case that these functions suggest the possibility of entity existence, they provide the reason of this suggestion.

The outcome of semiologic exploratory functions are seen in the Report.

These functions are activated only if in Introduction the user allows DS to make decisions.

### Focality criteria

In Digital Semiology, focality is defined according to several criteria.

#### Unconditioned criteria:

1. Figure of 4 (Rt or Lt)
2. Fencer's posturing (Rt or Lt)
3. Forced head turn (Rt or Lt)
4. Forced gaze deviation (Rt or Lt)

#### Conditioned Criteria:

1. **Hypermotor** event (if started earlier than 5 sec before start of GTCS).
2. **Aura reporting** event (if started earlier than 5 sec before any other event, except of another aura reporting event or trigger event).
3. Unilateral **tonic** movement of arm or leg or side of face or hemibody (if started earlier than 5 second before start of contralateral tonic or dystonic or clonic movements or contralateral paralysis)
4. Unilateral **dystonic** movement of arm or leg or side of face or hemibody (if started earlier than 5 second before start of contralateral tonic or dystonic or clonic movements or contralateral paralysis)
5. Unilateral **clonic** movement of arm or leg or side of face or hemibody (if started earlier than 5 second before start of contralateral tonic or dystonic or clonic movements or contralateral paralysis)
6. Unilateral **paralysis** of arm or leg or side of face or hemibody (if started earlier than 5 second before start of contralateral paralysis).

### Motor jacksonian march criteria

1. Jacksonian march is defined as spatio-temporal clonic movement propagation on the same body side between face, arm, and leg and/or different parts of distal extremities.
2. If between two epochs with clonic movements is an epoch without clonic movements, such sequences are not considered as Jacksonian march.
3. If at the same moment leg and face are simultaneously involved in clonic movements, further propagation is not considered as Jacksonian march.

4. Synchronous bilateral clonic movements (with the same start and end timings) are excluded. However, if not completely synchronous, clonic movements involving both body sides are not excluded, rather considered as to separate sequences.

#### [PNES criteria](#)

1. At least part of time the eyes of patient were closed during ictal episode.
2. The patient resisted the attempt to open eyes by observer.
3. The patient completely controlled her/his fall during ictal episode.
4. The duration of ictal episode was longer than 5 minutes.

#### [SUDEP risk criteria](#)

1. GTCS started from sleep
2. GTCS ended on abdomen
3. After GTCS the patient turned to abdomen
4. Apnea after GTCS
5. Cyanosis after GTCS
6. GTCS (It is important to differentiate between GTCS only and GTCS plus additional SUDEP risk factors.)

## [EDITING ICTAL EPISODE](#)

After user finalized EVENTS part, DS continues to the next part, which is EDITING ICTAL EPISODE. The workflow of this part differs between General View Mode and Add-on Mode. Here we will first discuss the workflow of this part for Add-on Mode and then will enter some remarks regarding General View Mode. (User can see the DS workflow scheme.)

In Add-on Mode, after user described all events, she/he should press the Next step button, and DS will proceed to the next step, which is Event removal.

#### [Event removal](#)

At this step, DS presents to user writing GUI and asks user to write the number of event (according to report) that should be removed.

## Writing GUI

|

Yoni, if you want to remove the reported event, write the event number, which should be removed.  
Otherwise, write 0. At the end press this button

If User writes 0, DS proceeds to next step, which is Editing reported event.

If user writes number that exceeds the number of events in the report, DS will generate error message and will return the same writing GUI.

If user writes existing event number, this event will be removed both from ictus and from the report and the events numeration in the report will be changed appropriately. Thereafter DS will again present the same GUI until user will write 0, then DS will proceed to the next step.

### [Editing reported event](#)

Here user is asked first to select the event that should be removed, but its content should be changed.

Which reported event should be edited if any?

## Writing GUI

|

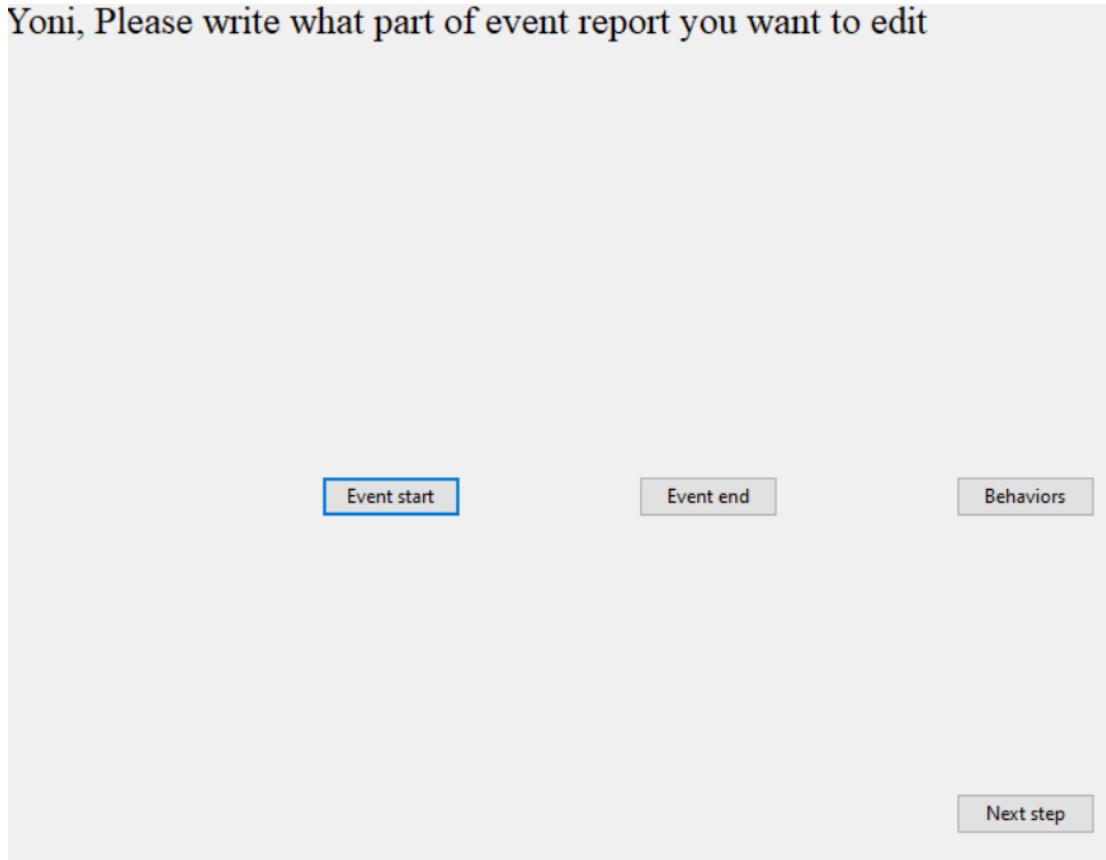
Yoni, if you want to edit the reported event, write the event number, which should be revised in the row above.  
Otherwise, write 0. At the end press this button

Here the user is asking to write which reported event should be edited. If user writes 0, no reported event will be edited, and DS will continue to the next step – Edit introduction.

If user writes number that exceeds the number of events in the report, DS will generate error message and will return the same writing GUI.

If user writes existing event number, DS will ask her/him, which part of reported event she/he wants to edit: timing of event start, timing of event end and/or behaviors, the user can select more than one option.

Yoni, Please write what part of event report you want to edit



If user selects Event start and /or Event end, DS will open appropriate GUIs. If user will indicate timing of Event start later than Event end, DS will ask user to redefine timings.

If user selects Behaviors, then DS will open GUIs of behaviors.

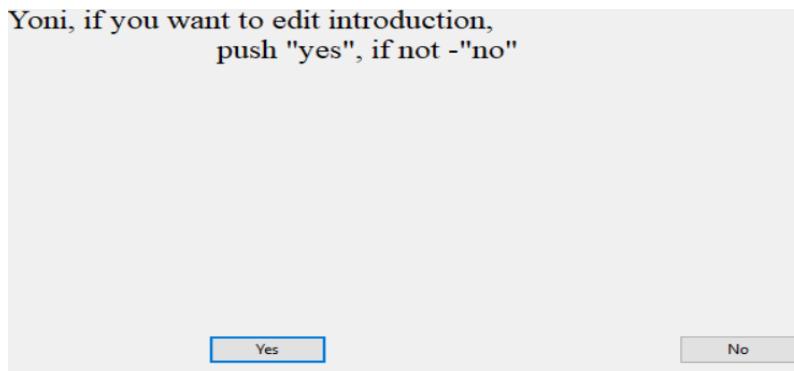
If user will press Next step button on the GUI presented just above or in the GUI of behaviors, DS will present the writing GUI asking to write the event number that should be edited or 0.

If user selects 0, DS will proceed to the next step.

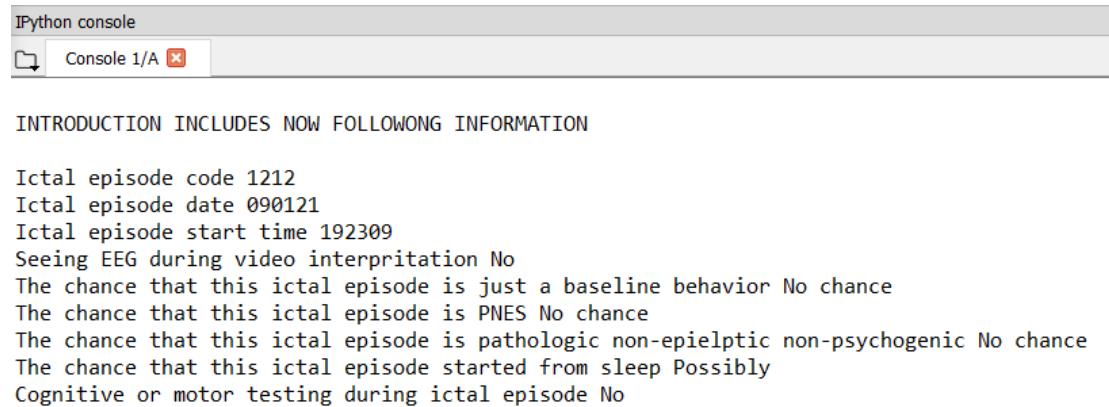
#### [Editing introduction](#)

At this step, DS opens the selectin GUI asking user whether she/he wants to edit introduction.

Yoni, if you want to edit introduction,  
push "yes", if not -"no"



At the same time In IPython Console appears the information, which was saved in INTRODUCTION. For example:



```
INTRODUCTION INCLUDES NOW FOLLOWONG INFORMATION

Ictal episode code 1212
Ictal episode date 090121
Ictal episode start time 192309
Seeing EEG during video interpritation No
The chance that this ictal episode is just a baseline behavior No chance
The chance that this ictal episode is PNES No chance
The chance that this ictal episode is pathologic non-epileptic non-psychogenic No chance
The chance that this ictal episode started from sleep Possibly
Cognitive or motor testing during ictal episode No
```

This information should help to user to decide whether or not to edit INTRODUCTION.

If user selects Yes, DS moves back to beginning of INTRODUCTION – to the question about ictal episode code, then DS bypasses parts of INTRODUCTION and come to ictal episode date (if ictal episode code did not change) and next DS pass through steps of INTRODUCTION to Motor/cognitive testing. Thereafter DS "jumps" to the next step after Editing introduction – Edit once again. (See DS workflow.)

During the INTRODUCTION editing, DS presents in IPython Console, how the given parameter was defined before editing. For example:

The ictal episode code was defined as 1212

or

The ictal episode date was defined as 090121

If user selects No in the GUI of Edit introduction, DS continues to the next step – Edit once again.

[Edit once again](#)

Here DS presents selecting GUI, asking user to select Yes or No regarding the editing the report once again.

Yoni, if you want to edit introduction,  
push "yes", if not -"no"

Yes

No

Whatever answer user will select, DS will continue to the next step – Postictal. However, the selection at this step will influence the workflow after Postictal step: if user selected No, DS will continue to offering user to write the final comment; if user selected Yes – DS will come back to Events and will go through whole EDITING ICTAL EPISODE part to Edit once again step.

The next step is Postictal.

#### [Postictal](#)

This step consists out of two stages:

The first stage is the question whether some of reported events can be postictal.

The second stage is relevant if user select Yes at the first one and it is – which event can be labeled as postictal.

At the first stage DS opens selecting GUI:

Yoni, if it is possible that some of the events represent postictal state,  
push "yes", if not -"no"

Yes

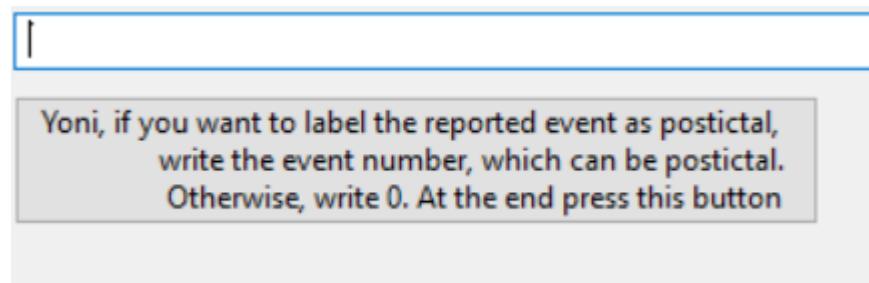
No

If user selects No, DS continues to the next step – Writing the final comment.

If user selects Yes, the following sentence is written to the report:

It is possible that last event\s represent(s) postictal state. This, however, can not be defined without taking into account EEG

and DS continues to the second stage of Postictal step – opens the writing GUI:



If user writes 0, DS continues to the next step – Writing the final comment.

If user writes the number, which exceeds the number of events, DS will return the GUI.

If user writes the existing event number, DS will write following sentence to the report (if for example, event number 1 was selected):

The event number 1 was labeled  
as possible postictal event.

Then DS will again present the same GUI, until the user will write 0.

#### Writing final comment

This step follows Postictal step. DS presents selecting GUI, asking whether the user wants to write final free text comment.

Yoni, if you want to add final free text comment to the report,  
push "yes", if not -"no"

Yes

No

If user selects No, DS will continue to the next step – DS exit.

If user selects Yes, DS generates the following sentence in IPython consol:

Software:

Yoni, please, write final free text comment and, at the end press Enter

The user can now write her/his comments in the IPython console and this comment will be included into report. After user will press enter, DS will continue to the next step DS exit. The new file or changes to the old file will be saved.

[DS exit](#)

Here the user is asked to press button on the GUI and exit DS.

---

 Digital Semiology exit

Yoni, thank you for finalizing ictal episode interpretation  
Please, push the button "Exit" to close Digital Semiology.

[Exit from Digital Semiology](#)

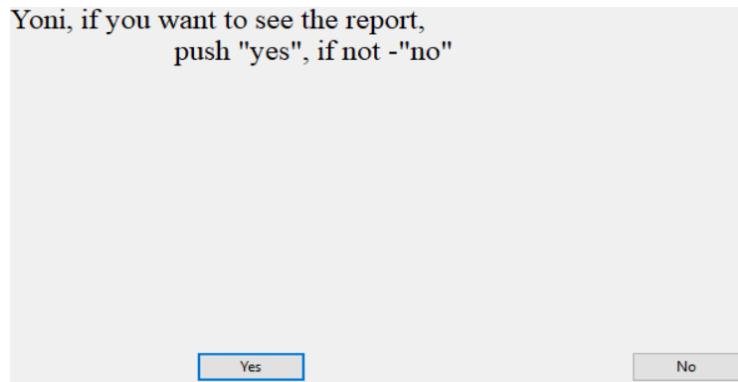
[The nuances of ictal episode editing in General view mode](#)

In previous sections we discussed the ictal episode editing in Add-on mode, the same process in General view mode is similar, but there are some differences. User can

see the DS workflow scheme. Most of the steps of ictal episode editing in General view mode are the same as in Add-on mode, however the three initial stems are different.

### *1. See the report*

After the user finalized describe the events, DS opens the selecting GUI. The user is offered to select, whether she/he wants to see the report.

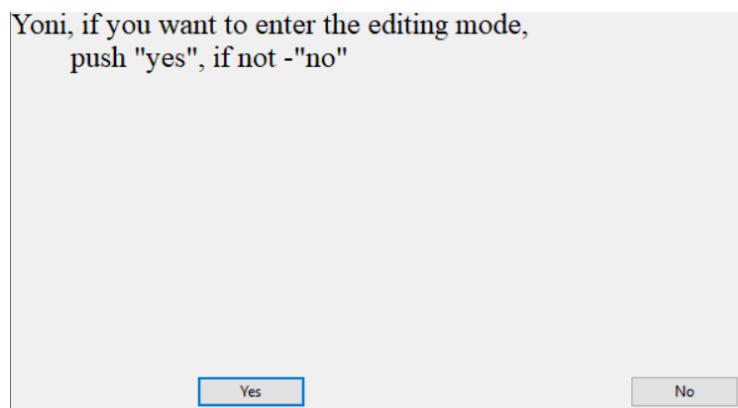


With any answer, DS will continue to the next step – Editing mode.

If user selected Yes, DS will present report in the IPython console, if No – the report will be not generated at this step (but will be generated further).

### *2. Editing mode*

Here DS asks user whether she/he wants to edit the ictal episode



If user answers No, DS bypasses the steps of editing ictal episode and come directly to Writing final comment step.

If user answers Yes, DS proceeds to Event addition step.

### *3. Event addition*

This step is identical to EVENTS part of Add-on mode.

# DIGITAL SEMIOLOGY REPORT

## DS Report: Overview

DS report describes ictal events and behaviors. The parts of introduction (except of ictal episode date) are not included into report. These parts, however, are encoded in ictus.

The events in the report are listed and numerated in chronological order according to time of event start.

The timings in the report are defined according to the time scale of video-player, which starts as 0 at the beginning of video.

DS report includes several parts:

1. Short description of ictal episode.
2. Detailed description of ictal episode.
3. Statements regarding postictal state.
4. Final comment.

### Short description of ictal episode

This part of report includes ictal episode date and list of events including their timings.

For example:

## DIGITAL SEMIOLOGY REPORT

The date of this ictal episode is 11.11.2011.

This ictal episode includes following events:

Event number 1 00:36:31-00:37:20 dialeptic behavior event  
Event number 2 00:36:48-00:37:25 motor automatism event  
Event number 3 00:37:42-00:38:41 generalized tonic-clonic event  
Event number 4 00:38:41-00:39:51 dialeptic behavior event

### Detailed description of ictal episode

This part of the report includes list of events and behaviors.

For example:

The detailed description of the ictal episode:

EVENT NUMBER 1 00:36:31-00:37:20 dialeptic behavior event  
This event includes following behaviors:

1)Partial unresponsiveness.

EVENT NUMBER 2 00:36:48-00:37:25 motor automatism event  
This event includes following behaviors:

1)semipurposeful movements involving:  
left arm (mild)

EVENT NUMBER 3 00:37:42-00:38:41 generalized tonic-clonic event  
This event includes following behaviors:

1) GTCS ended in the position on the back.  
After GTCS the patient turned to position on the back.  
The time of transition from tonic to clonic phase of GTCS is 00:38:13.

EVENT NUMBER 4 00:38:41-00:39:51 dialeptic behavior event  
This event includes following behaviors:

1)Complete unresponsiveness.

#### Statements regarding postictal state

Here are the statements: a) whether it is possible that some of reported events are postictal and b) which events user labeled as possible postictal.

For example:

It is possible that last event\s represent(s) postictal state. This, however, can not be defined without taking into account EEG

The event number 4 was labeled  
as possible postictal event.

#### Final comment

If user wrote the final comment, it will be presented in this part of report.

For example:

---

#### FINAL FREE TEXT COMMENT

It is hard to make a conclusion without EEG!

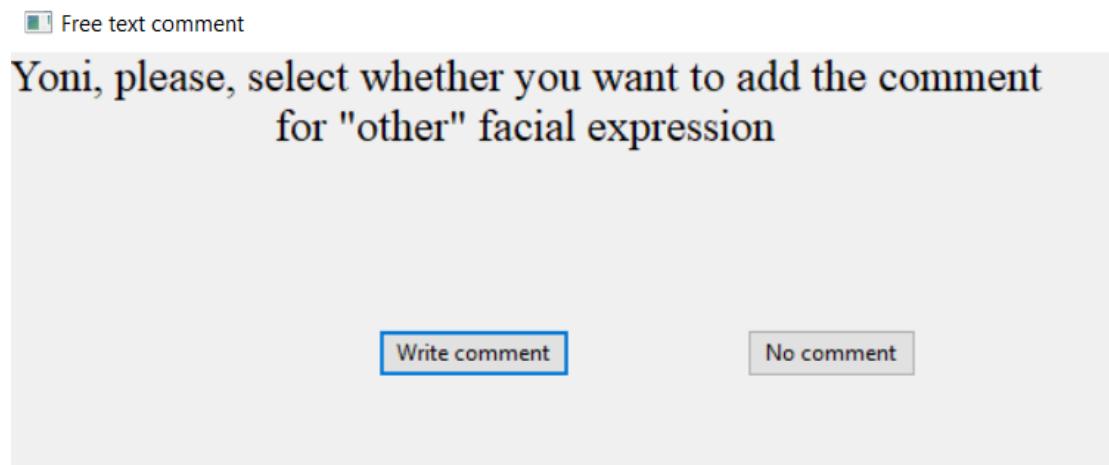
#### Writing free text comments in DS

Final comment is only one of the ways to write free text comments in DS. Another opportunity to write comment is when user defines the behavior as "other". In DS it

is a difference between "other" behavior and "unspecified" behavior. "Other" behavior is one that can be described in specific terms, but these terms do not exist in the present version of digital semiology. In contrast, "unspecified" behavior is difficult to describe in specific terms. When user selects "unspecified" behavior, DS does not open possibility to write comment.

For example, if user wants to describe the face expression of patient as "thinking" but this specific term is absent in DS, then user can select "other" face expression and add the comment "thinking". In another case, when user can not specify the face expression in existing terms of DS but cannot imagine any specific term to describe this behavior, she/he can select "unspecified" face expression – then DS will not open the possibility to write comment.

When user selects "other" behavior, DS will open selecting GUI presenting three choices, for example:



If user selects "Write comment ", DS will open the writing GUI:



After user wrote the comment, DS will continue to the next step.

If user selects "No comment", DS will continue to the next step, without opening possibility to write comment.

The final comment before exit from the program DS allows to write in the console only.

## EXIT FROM DS

After finalizing ictal episode encoding, DS bring the user to the point when she/he should exit DS (see paragraph DS exit). In this case all changes will be saved.

It is possible also to exit from DS without going through all the way to the end. In the GUI of Events in Add-on mode (the same GUI in Add event step in General view mode). In this GUI if user selects Exit button, DS will ask the user whether she/he wants to save the changes – DS will open the GUI:

Yoni, please select: whether or not you want to save your input to the ictal episode interpretation during this session

Save

Do not save

If user selects "Do not save", the changes of the session will not be saved, and DS will present the following GUI:

Digital Semiology exit

Yoni, the new file was NOT saved,  
thank you for finalizing ictal episode interpretation  
Please, push the button "Exit" to close Digital Semiology.

Exit from Digital Semiology

If user selects "Save", the changes will be saved. In this case DS opens the following GUI:

Yoni, the new file was saved, thank you for finalizing ictal episode interpretation  
Please, push the button "Exit" to close Digital Semiology.

[Exit from Digital Semiology](#)

Presenting words "new file" DS means here either the changes to old file or the new file created during the session.

In addition, it is the possibility to exit from DS using GUIs in INTRODUCTION. Some GUIs have button: "Exit without saving" and other GUIs – Exit (In this case DS will open the GUI asking the user, whether to save the new file).

## THE PROGRAM UNPACK\_ICTAL\_EPISODE

DS creates .dat binary file. To create .txt file user should employ another python-based program – unpack\_ictal\_episode (UIE).

UIE creates both .txt file and graphical presentation of ictal episode.

UIE code should be placed in the same directory with DS code. It is possible to operate UIE both in Spyder and without Spyder (just by double click on the UIE icon).

When UIE is open, it writes in the console the following statement:

Please, write the code of this ictal episode and then press Enter

After the user writes (or pastes) the code of ictal episode into console and presses Enter, UIE reads the .dat binary file of the ictal episode (according to code, which is a part of .dat binary file name), UIE presents in the console the report and ictus (encoded values) of this ictal episode.

UIE also sometimes displays some warning message from Matplotlib module,

For example:

```

Unrecognized location "".
Falling back on "best"; valid locations are
best
upper right
upper left
lower left
lower right
right
center left
center right
lower center
upper center
center

% (loc, '\n\t'.join(self.codes)))

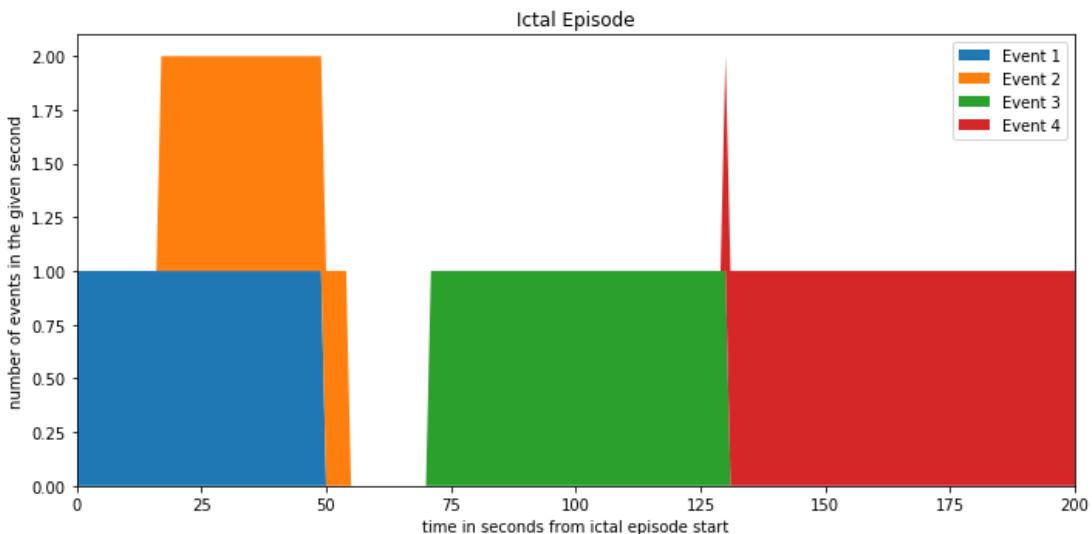
```

We recommend ignoring this message.

At the end, UIE displays in the console following statement:

**Press Enter to exit**

When user press Enter, the UIE is closed and the graphical presentation of ictal episode (GPIE) appears in the console, for example:



GPIE is created by Matplotlib module. Different events are shown by different colors. The correspondence between colors and the event numbers are shown in right upper angle of the GPIE. The details about events user can find in the report.

The x-axis of GPIE is a time in seconds from the beginning of video (the same timescale as in the report). If the events overlap in time, then one of the events is shown on the top of another during the overlapping epoch. The y-axis is number of events in a given timepoint (in a given second). We recommend ignoring values that are not integers in the y-scale.

To save the GPIE we recommend using Snipping tool:

<https://support.microsoft.com/en-us/windows/use-snipping-tool-to-capture-screenshots-00246869-1843-655f-f220-97299b865f6b>

The .text file describing the ictal episode will be found in the same directory as UIE after exit from UIE.

## THE PROGRAM DS\_VERSION\_ADAPTER

User can modify the code of DS. For example, she/he can add behaviors to the events. This can enlarge the lists of behaviors in ictus. Therefore, new versions can be inappropriate for editing of the files created by old versions.

To avoid this problem, user can apply DS\_version\_adapter, which is Python code, able to convert the .dat files created by old DS version to be readable by new version.

The authors recommend updating DS\_version\_adapter, so that the length of list of behaviors correspond to new DS version.

```

#This code adapts .dat files to version 07.02.2021 of Digital Semiology
import pickle
ictal_episode_code=input('Please, write the code of this ictal episode a
DS_version=input('Please, write the version of Digital Semiology and the
filename='ictal_episode_' + ictal_episode_code +'.dat'
f=open(filename, 'rb')
ds_starter = pickle.load(f)
ictus=pickle.load(f)
report=pickle.load(f)
software_user_dialogue=pickle.load(f)
f.close()
name_changer=0
for i in range (0,len(ictus)):
    for j in range (4,len(ictus[i])):
        if ictus[i][0]==0: #Simple motor
            if len(ictus[i][j])<52:
                ictus[i][j]=ictus[i][j]+['']* (52-len(ictus[i][j]))
                name_changer=1
        if ictus[i][0]==1: #Automatisms
            if len(ictus[i][j])<12:
                ictus[i][j]=ictus[i][j]+['']* (12-len(ictus[i][j]))
                name_changer=1
        if ictus[i][0]==2: #Autonomic
            if len(ictus[i][j])<4:
                ictus[i][j]=ictus[i][j]+['']* (4-len(ictus[i][j]))
                name_changer=1
        if ictus[i][0]==3: #Eye movements
            if len(ictus[i][j])<23:
                ictus[i][j]=ictus[i][j]+['']* (23-len(ictus[i][j]))
                name_changer=1
        if ictus[i][0]==4: #Hyperkinetic
            if len(ictus[i][j])<12:
                ictus[i][j]=ictus[i][j]+['']* (12-len(ictus[i][j]))

```

The red arrows show the numbers that can be changed if in the new DS version new behaviors are added.

After the code is activated by pressing Run button, the program will present to user the following instruction in IPython console:

**Please, write the code of this ictal episode and then press Enter**

After user wrote ictal episode code and pressed enter, the program will present the second instruction in IPython console:

**Please, write the code of this ictal episode and then press Enter**

After that, if needed, the program will modify ictus variable in the .dat file, adapting it to the current DS version.

If ictus variable in .dat file was changed, the program will rename the .dat file. For example, if the old file was ictal\_episode\_1212.dat, the new file will be

`ictal_episode_1212_vers07.02.2021.dat` (if .dat file was adapted to the version 07.02.2021 of DS).

If the .dat file corresponds to the current DS version, then `DS_version_adapter` will generate in IPython console the following statement:

`ictal_episode_1212.dat` doesn't require adaptation to the current Digital Semiology version

If user indicated ictal episode code of non-existing .dat file, DS will generate the following error message:

```
FileNotFoundException: [Errno 2] No such file or directory: 'ictal_episode_12.dat'
```