



Human Centered Multimedia
Institute of Computer Science

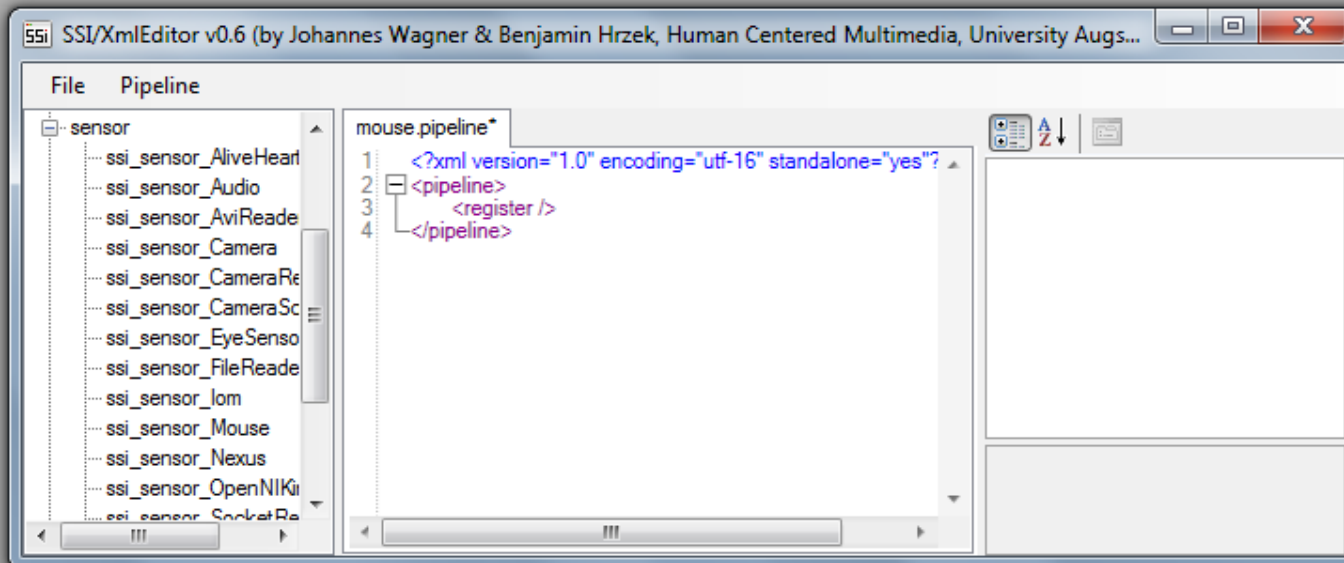


Social Signal Interpretation XML Tutorial

Johannes Wagner <wagner@openssi.net>
(last update: 16.01.13)

<http://openssi.net>

- Start `bin/<Win32|x64>/vc10/xmledit.exe`
- Create a new pipeline (File>New) and save it (File->Save), e.g. `mouse.pipeline`

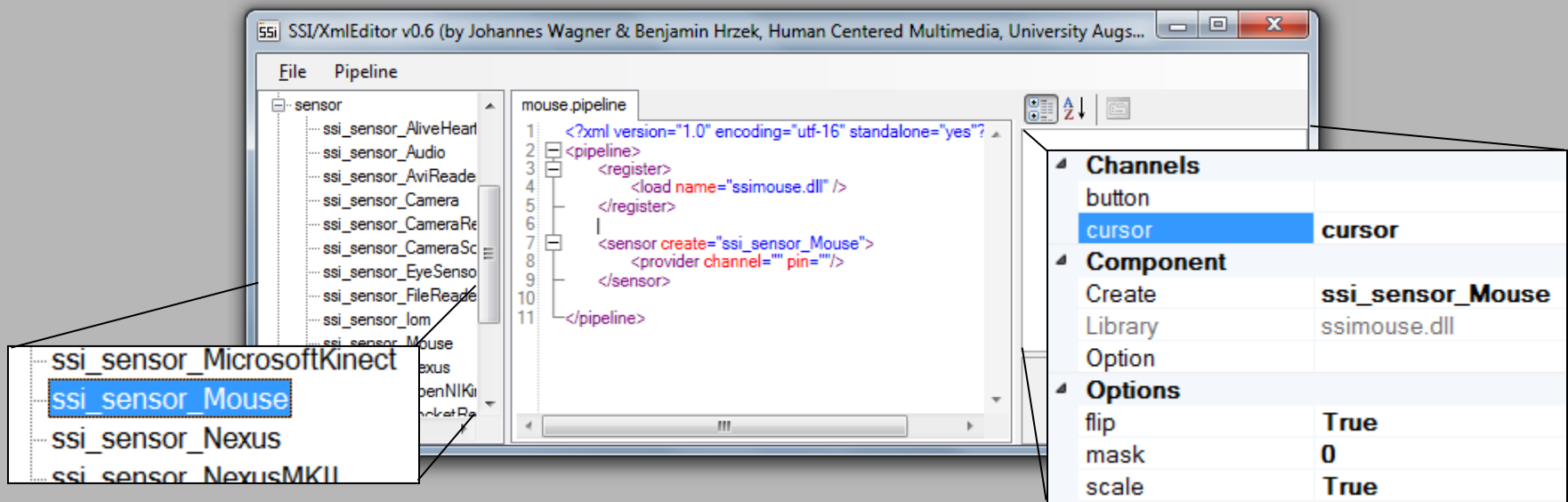


components

editor

properties

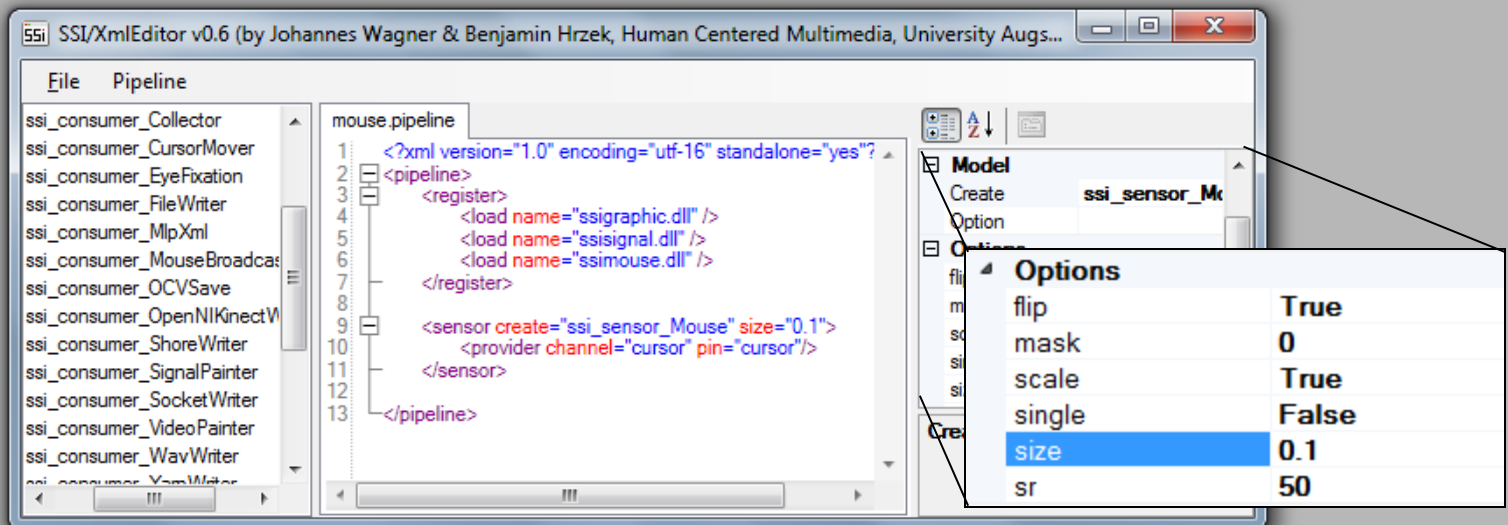
- Within the component panel expand the **sensor**-tree and double click **ssi_sensor_Mouse**
- Place cursor in line `<sensor...` to display the properties
- Below **Channels** add a pin-name to **cursor** (e.g. also **cursor**)
- Input components can now receive the cursor stream via the pin-name



The screenshot shows the SSI/XmlEditor v0.6 interface. On the left, the 'File Pipeline' panel shows a tree of sensors, with **ssi_sensor_Mouse** selected. The main area displays the XML pipeline for 'mouse.pipeline', which includes a `<sensor create="ssi_sensor_Mouse" provider channel="" pin="" />` tag. On the right, the 'Channels' panel is open, showing a table with the following data:

Channels	
button	
cursor	cursor
Component	
Create	ssi_sensor_Mouse
Library	ssimouse.dll
Option	
Options	
flip	True
mask	0
scale	True

- Below the category **Channels** you find options specific to the component
- Additional information to a selected option is displayed at the bottom of the panel
- Change option **size** to **0.1**



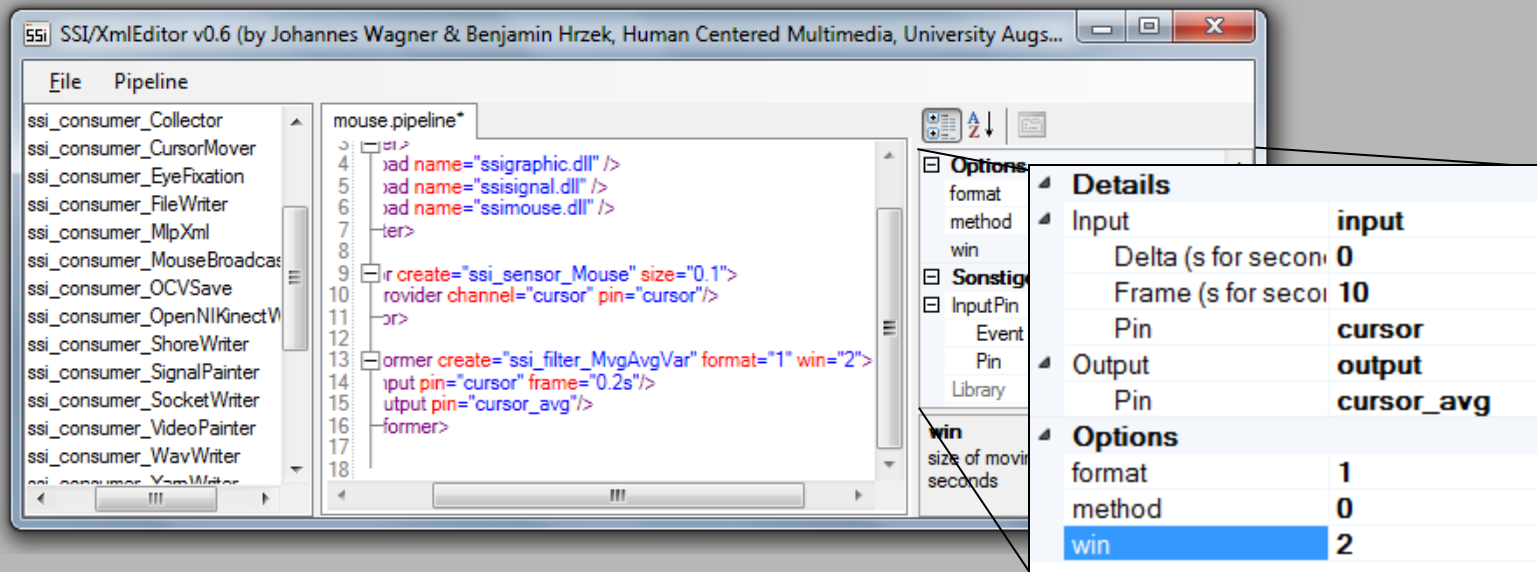
The screenshot shows the SSI/XmlEditor v0.6 interface. The main window displays the 'mouse.pipeline' file with the following XML content:

```
<?xml version="1.0" encoding="utf-16" standalone="yes"?>
<pipeline>
  <register>
    <load name="ssigraphic.dll" />
    <load name="ssisignal.dll" />
    <load name="ssimouse.dll" />
  </register>
  <sensor create="ssi_sensor_Mouse" size="0.1">
    <provider channel="cursor" pin="cursor"/>
  </sensor>
</pipeline>
```

The 'Options' panel on the right shows the 'ssi_sensor_Mouse' component. The 'Options' table is displayed below the component name:

Options	
flip	True
mask	0
scale	True
single	False
size	0.1
sr	50

- Place cursor after `</sensor>` and insert a **MvgAvgVar** Filter (available from the transformer tree)
- In the properties panel set the input pin to **cursor** and choose a name for the output pin (here **cursor_avg**)
- Set frame size either by samples per second (e.g. "10") or in seconds (z.B. "0.2s")
- Set options **format** to **1** and **win** to **2**



The screenshot shows the SSI/XmlEditor v0.6 interface. The main window displays the XML pipeline for 'mouse.pipeline'. The pipeline includes a sensor and a filter. The filter is configured with the following attributes:

```

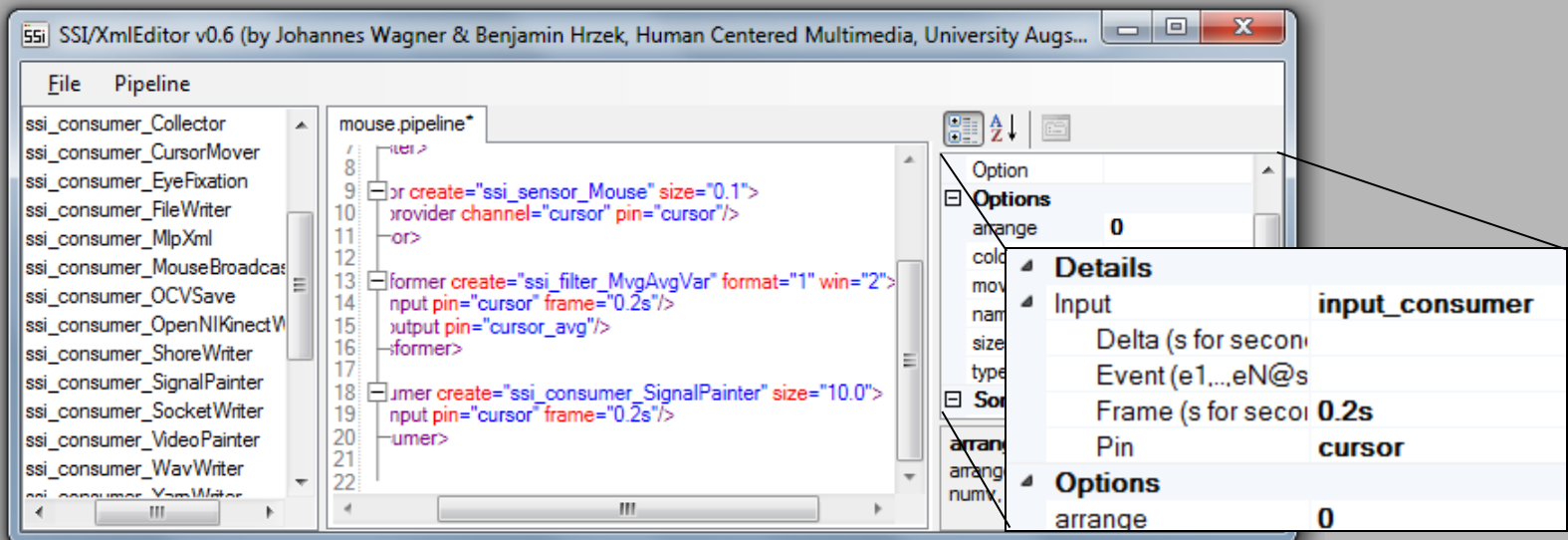
<filter create="ssi_filter_MvgAvgVar" format="1" win="2">
  <input pin="cursor" frame="0.2s"/>
  <output pin="cursor_avg"/>
</filter>

```

The right-hand side of the interface shows the 'Options' panel for the selected filter. The 'Details' tab is active, showing the following configuration:

Details	
Input	input
Delta (s for second)	0
Frame (s for second)	10
Pin	cursor
Output	output
Pin	cursor_avg
Options	
format	1
method	0
win	2

- Place cursor after `</transformer>` and add a **SignalPainter** consumer
- Set input pin to **cursor** and choose a frame size (e.g. **0.2s**)
- Set option **size** to **10.0**
- Execute pipeline by pressing **F5**



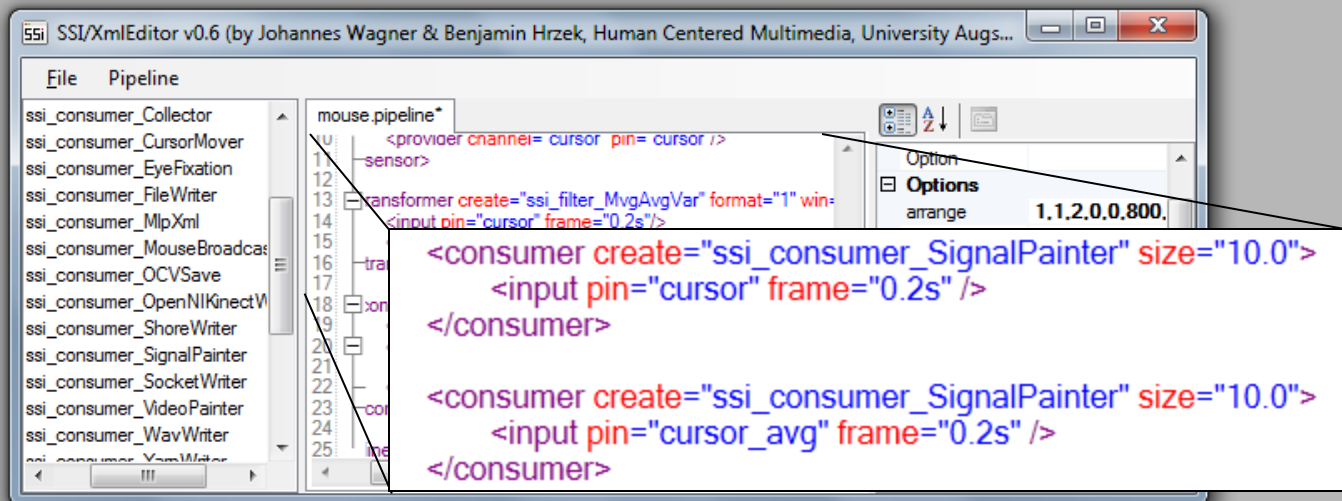
The screenshot shows the SSI/XmlEditor v0.6 interface. The main window displays the XML pipeline configuration for 'mouse.pipeline'. The pipeline consists of three components: a sensor, a filter, and a consumer. The consumer is a 'SignalPainter' with the following configuration:

```
<consumer create="ssi_consumer_SignalPainter" size="10.0">
  <input pin="cursor" frame="0.2s"/>
</consumer>
```

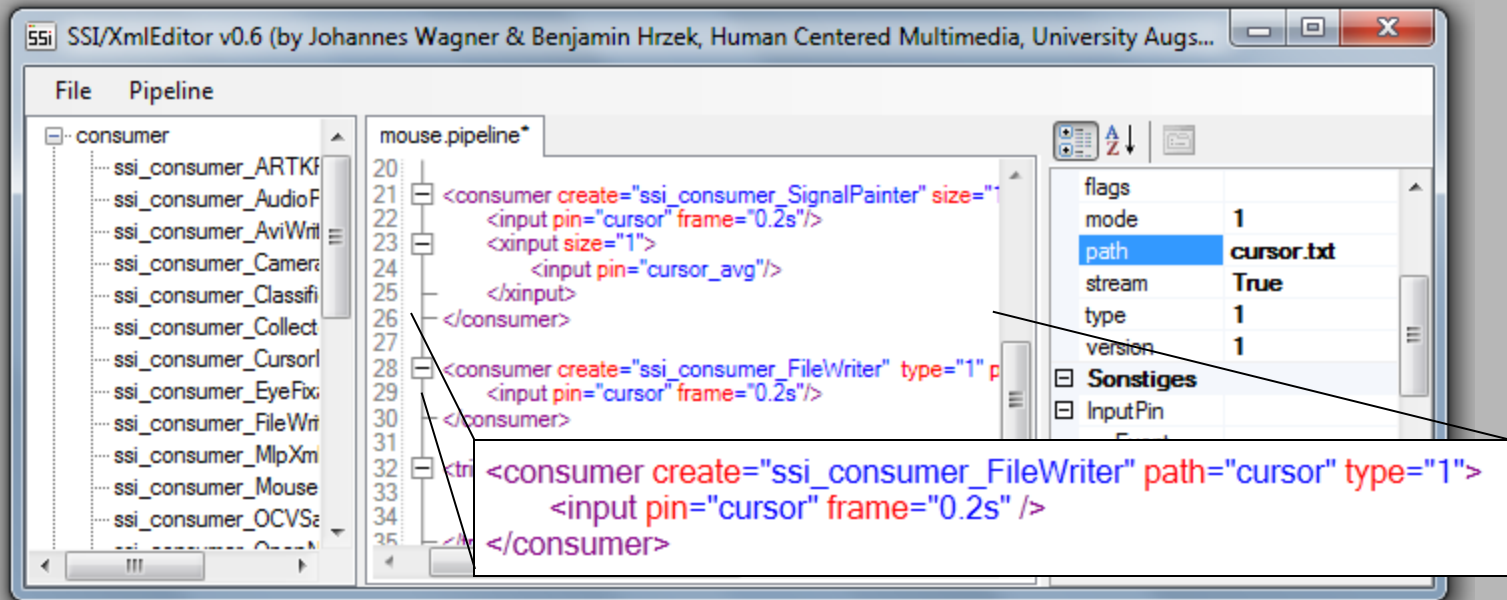
The 'Details' panel on the right shows the configuration for the selected 'SignalPainter' component:

Property	Value
Input	input_consumer
Delta (s for second)	
Event (e1,...,eN@s)	
Frame (s for second)	0.2s
Pin	cursor
Options	
arrange	0

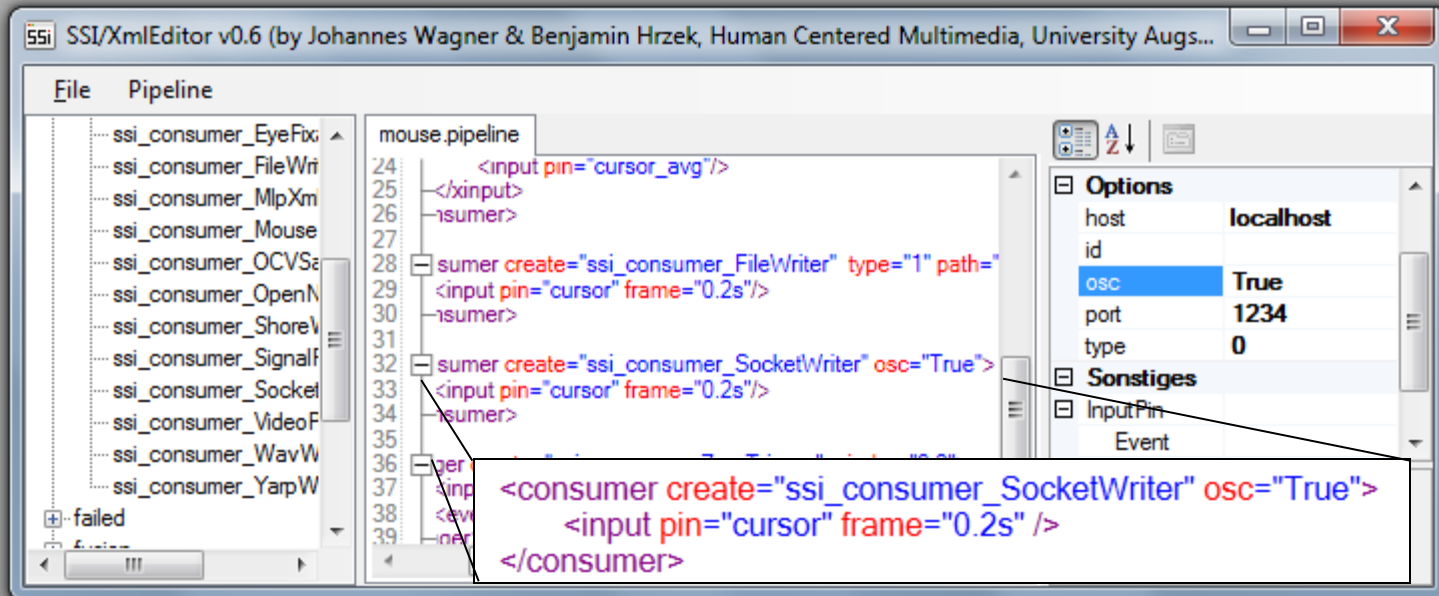
- To compare the raw **cursor** signal with its manipulated version either insert another **SignalPainter** consumer with input pin **cursor_avg**.
- Set option **arrange** to **1,1,2,0,0,400,600**



- To store **cursor** signal to disk add a **FileWriter** consumer and set once more the pin and frame property
- Use option **path** to set a file path and change **mode** to **1** (=ASCII)
- When you run the pipeline again the cursor signal will be stored in files **cursor.stream** and **cursor.stream~**



- To stream the **cursor** signal through a socket insert a **SocketWriter** consumer and set option **osc** to **True**
- Start pipeline and run the following command on the command line:
`<root>\[Win32|x64]\vc10\bin\sockspy.exe --osc -console 1234`

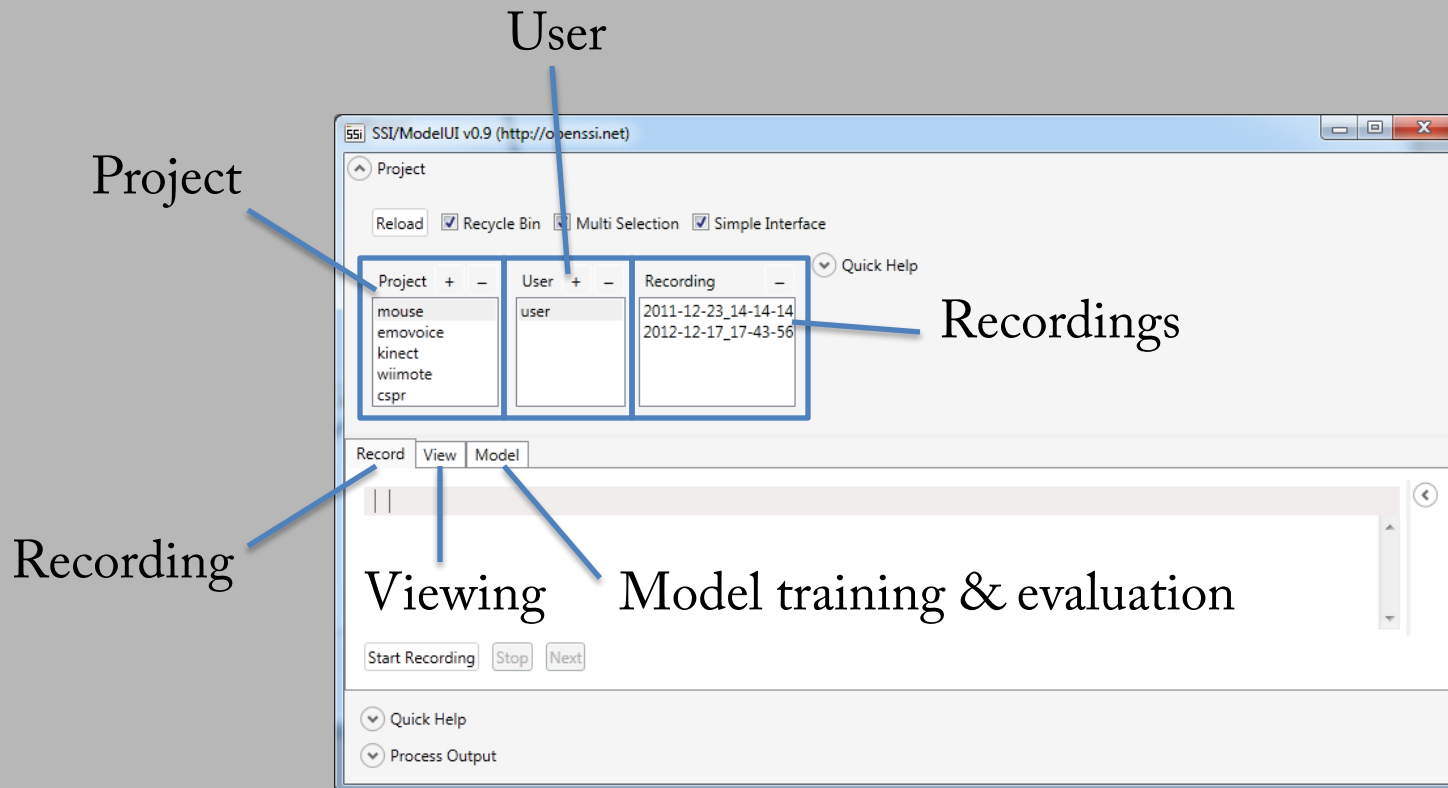




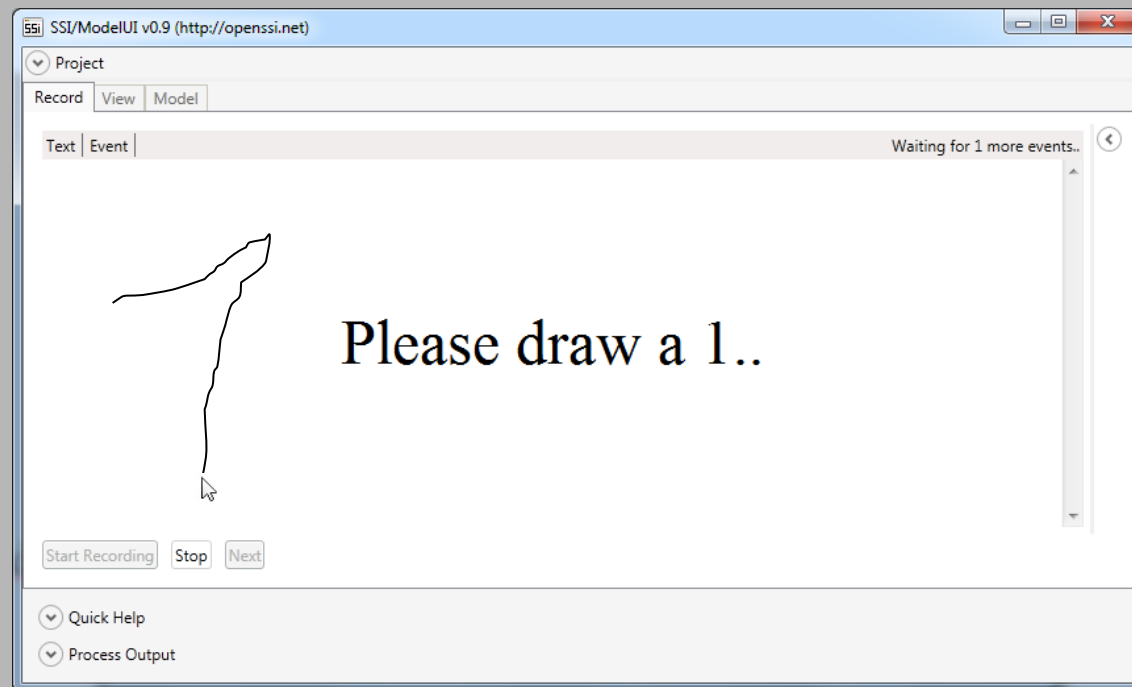
ModelUI



- Start `<root>\[Win32|x64]\vc10\bin\modelui.exe` and select project `mouse`



- Select **user** and click **Start Recording** in record panel
- Follow instructions on screen
- Draw inside the GUI by holding the left mouse button pressed



- Switch to **Model** panel and switch to **Train/Run**
- Select one or more **recordings** (by holding the Strg key)
- Select **dollar** as training method panel
- Press **Train Model** button

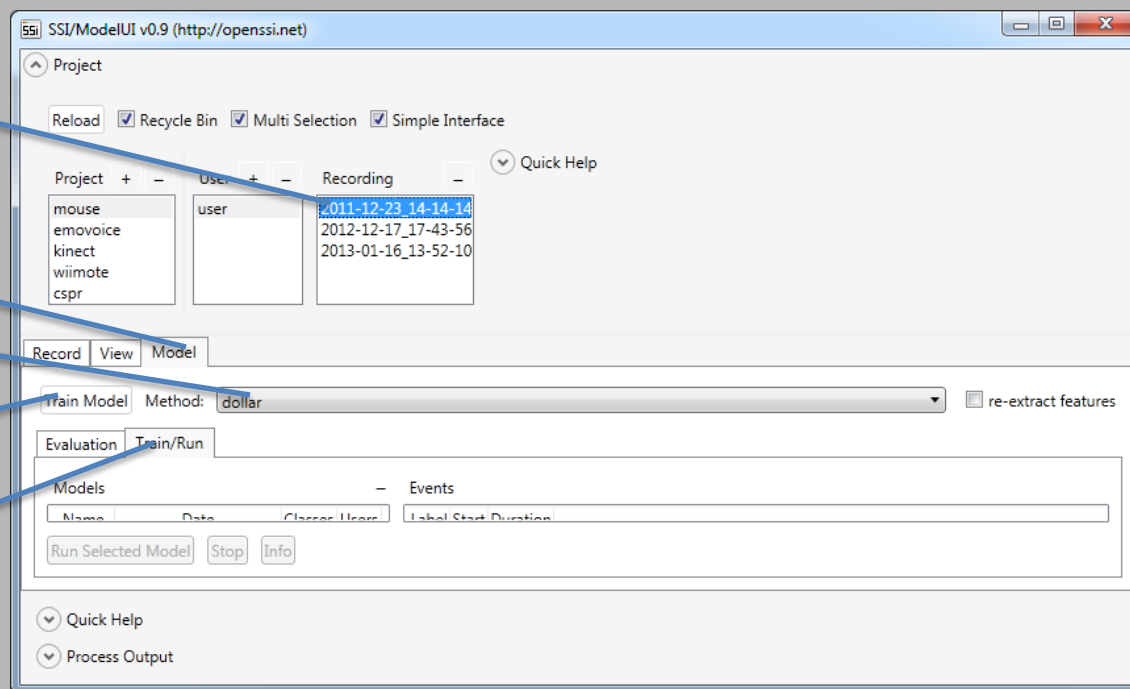
Choose Recording(s)

Switch to Model

Select Method

Train Model

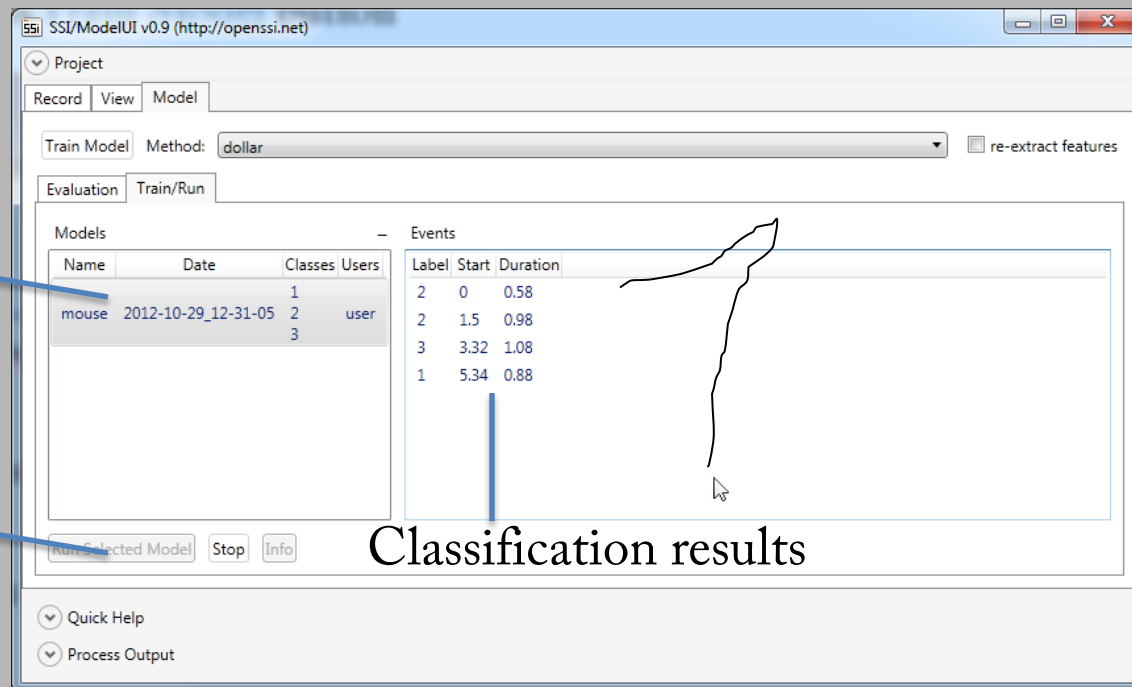
Switch to
Train/Run



- Select the trained **model** and click **Run Selected Model**
- Draw inside the GUI by holding the left mouse button pressed

Choose Model

Run Model



SSI/ModelUI v0.9 (<http://openssi.net>)

Project

Record View Model

Train Model Method: **dollar** ☐ re-extract features

Evaluation Train/Run

Models			
Name	Date	Classes	Users
mouse	2012-10-29_12-31-05	1 2 3	user

Events		
Label	Start	Duration
2	0	0.58
2	1.5	0.98
3	3.32	1.08
1	5.34	0.88

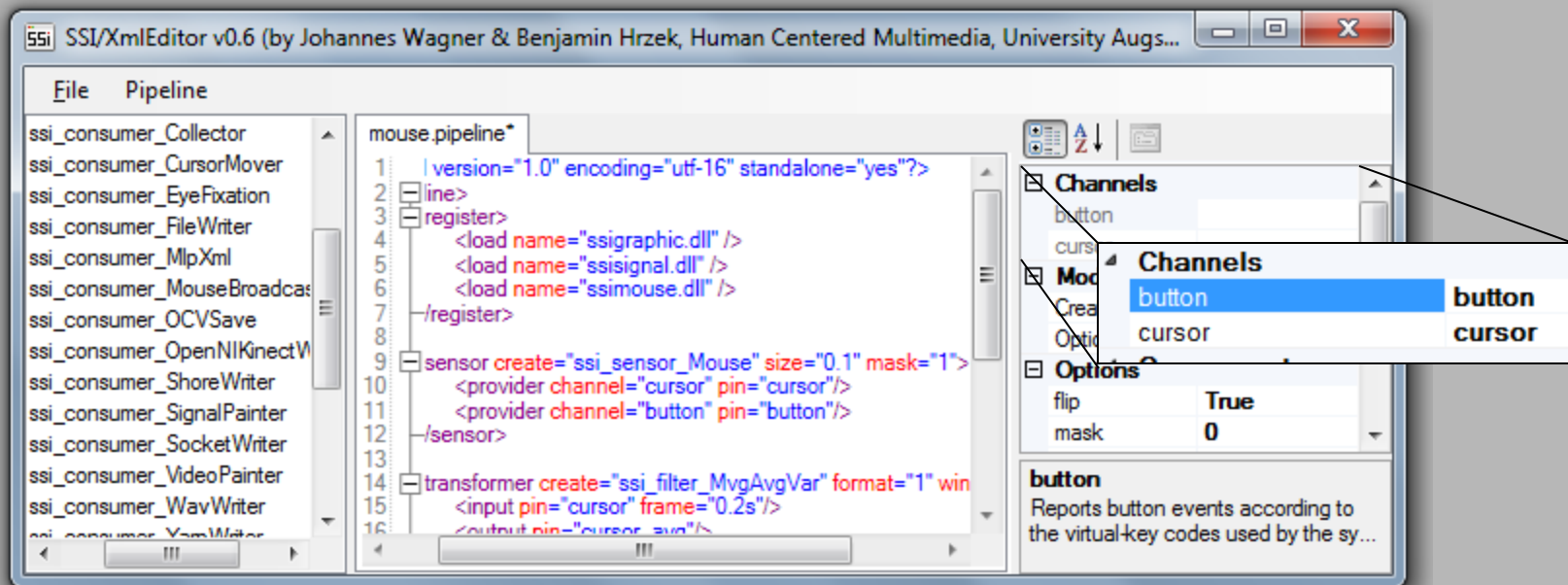
Run Selected Model Stop Info

Quick Help

Process Output

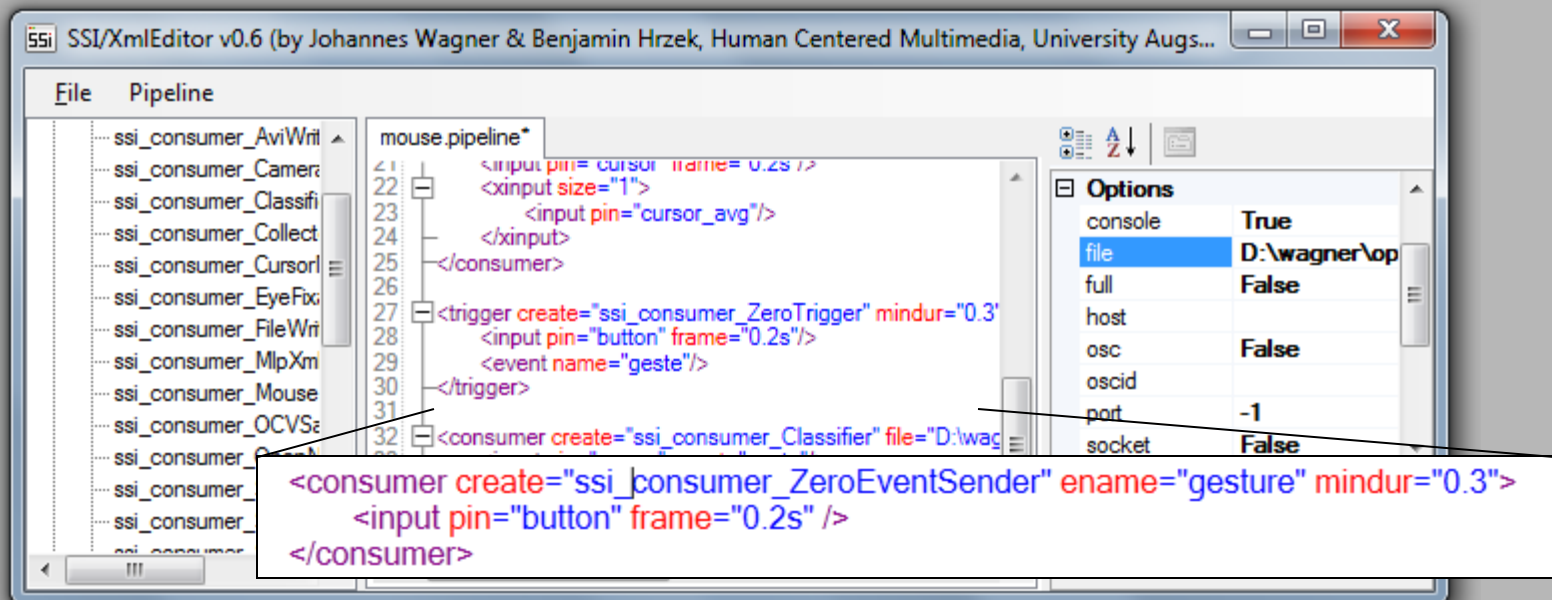
Classification results

- Switch back to the editor and place the cursor again in the `<sensor...` line
- Set `mask` of mouse sensor to `1` and activate second provider by putting a name to the `button` channel name (e.g. `button`)



Trigger

- Add a new consumer **ZeroEventSender** with input pin **button** and a frame size of **0.2s**
- Input an event name by setting option **ename** (e.g. **gesture**) and set option **mindur** to **0.3**



Classifier

- Insert a consumer of type **Classifier** and set input to **cursor**
- Instead of a frame size set **gesture@** as **Event** name
- As option **trainer** set the path of the previously trained model (e.g.: `<root>\model\mouse\train\2011-07-20_07-57-53\mouse`) to complete your online classifier

