In [1]:

```
# import statements
from time import sleep
from kafka import KafkaConsumer
import datetime as dt
import matplotlib.pyplot as plt

# this line is needed for the inline display of graphs in Jupyter Notebook
%matplotlib notebook

topic = 'Week9-Topic'

def annotate_max(x, y, ax = None):
    ymax = max(y)
    xpos = y.index(ymax)
    xmax = x[xpos]
    text = 'Max: Time={}, Value={}'.format(xmax, ymax)
    if not ax:
        ax=plt.gca()
    ax.annotate(text, xy=(xmax, ymax), xytext=(xmax, ymax+5),
arrowprops=dict(facecolor='red', shrink=0.05),)

def annotate_min(x, y, ax = None):
    ymin = min(y)
    xpos = y.index(ymin)
    xmin = x[xpos]
    text = 'Min: Time={}, Value={}'.format(xmin, ymin)
    if not ax:
        ax=plt.gca()
    ax.annotate(text, xy=(xmin, ymin), xytext=(xmin, ymin+5),
arrowprops=dict(facecolor='orange', shrink=0.05),)

def connect_kafka_consumer():
    _consumer = None
    try:
         _consumer = KafkaConsumer(topic,
                                   consumer_timeout_ms=10000, # stop iteration if no
message after 10 sec
                                   auto_offset_reset='earliest', # comment this if
you don't want to consume earliest available message
                                   bootstrap_servers=['localhost:9092'],
                                   api_version=(0, 10))
    except Exception as ex:
        print('Exception while connecting Kafka')
        print(str(ex))
    finally:
        return _consumer

def init_plots():
    try:
        width = 9.5
        height = 6
        fig = plt.figure(figsize=(width,height)) # create new figure
        ax = fig.add_subplot(111) # adding the subplot axes to the given grid
position
```

```
        fig.suptitle('Real-time uniform stream data visualization with interesting
points') # giving figure a title
        ax.set_xlabel('Time')
        ax.set_ylabel('Value')
        ax.set_ylim(0,110)
        ax.set_yticks([0,20,40,60,80,100])
        fig.show() # displaying the figure
        fig.canvas.draw() # drawing on the canvas
        return fig, ax
    except Exception as ex:
        print(str(ex))

def consume_messages(consumer, fig, ax):
    try:
        # container for x and y values
        x, y = [], []
        # print('Waiting for messages')
        for message in consumer:
            data = str(message.value.decode('utf-8')).split(', ')
            x.append(data[0])
            y.append(int(data[1]))
            # print(y)
            # we start plotting only when we have 10 data points
            if len(y) > 10:
                ax.clear()
                ax.plot(x, y)
                ax.set_xlabel('Time')
                ax.set_ylabel('Value')
                ax.set_ylim(0,110)
                ax.set_yticks([0,20,40,60,80,100])
                annotate_max(x,y)
                annotate_min(x,y)
                fig.canvas.draw()
                x.pop(0) # removing the item in the first position
                y.pop(0)
        plt.close('all')
    except Exception as ex:
        print(str(ex))

if __name__ == '__main__':

    consumer = connect_kafka_consumer()
    fig, ax = init_plots()
    consume_messages(consumer, fig, ax)
```

```
Waiting for messages
[70]
[70, 2]
[70, 2, 84]
[70, 2, 84, 86]
[70, 2, 84, 86, 47]
[70, 2, 84, 86, 47, 53]
[70, 2, 84, 86, 47, 53, 89]
[70, 2, 84, 86, 47, 53, 89, 63]
[70, 2, 84, 86, 47, 53, 89, 63, 43]
[70, 2, 84, 86, 47, 53, 89, 63, 43, 45]
[70, 2, 84, 86, 47, 53, 89, 63, 43, 45, 84]
[2, 84, 86, 47, 53, 89, 63, 43, 45, 84, 40]
[84, 86, 47, 53, 89, 63, 43, 45, 84, 40, 35]
[86, 47, 53, 89, 63, 43, 45, 84, 40, 35, 75]
[47, 53, 89, 63, 43, 45, 84, 40, 35, 75, 26]
[53, 89, 63, 43, 45, 84, 40, 35, 75, 26, 26]
[89, 63, 43, 45, 84, 40, 35, 75, 26, 26, 16]
[63, 43, 45, 84, 40, 35, 75, 26, 26, 16, 65]
[43, 45, 84, 40, 35, 75, 26, 26, 16, 65, 77]
[45, 84, 40, 35, 75, 26, 26, 16, 65, 77, 31]
[84, 40, 35, 75, 26, 26, 16, 65, 77, 31, 65]
[40, 35, 75, 26, 26, 16, 65, 77, 31, 65, 83]
[35, 75, 26, 26, 16, 65, 77, 31, 65, 83, 23]
[75, 26, 26, 16, 65, 77, 31, 65, 83, 23, 6]
[26, 26, 16, 65, 77, 31, 65, 83, 23, 6, 75]
[26, 16, 65, 77, 31, 65, 83, 23, 6, 75, 66]
[16, 65, 77, 31, 65, 83, 23, 6, 75, 66, 4]
[65, 77, 31, 65, 83, 23, 6, 75, 66, 4, 51]
[77, 31, 65, 83, 23, 6, 75, 66, 4, 51, 12]
[31, 65, 83, 23, 6, 75, 66, 4, 51, 12, 16]
```

```
[65, 83, 23, 6, 75, 66, 4, 51, 12, 16, 38]
[83, 23, 6, 75, 66, 4, 51, 12, 16, 38, 30]
[23, 6, 75, 66, 4, 51, 12, 16, 38, 30, 28]
[6, 75, 66, 4, 51, 12, 16, 38, 30, 28, 21]
[75, 66, 4, 51, 12, 16, 38, 30, 28, 21, 18]
[66, 4, 51, 12, 16, 38, 30, 28, 21, 18, 63]
[4, 51, 12, 16, 38, 30, 28, 21, 18, 63, 4]
[51, 12, 16, 38, 30, 28, 21, 18, 63, 4, 54]
[12, 16, 38, 30, 28, 21, 18, 63, 4, 54, 83]
[16, 38, 30, 28, 21, 18, 63, 4, 54, 83, 98]
[38, 30, 28, 21, 18, 63, 4, 54, 83, 98, 74]
[30, 28, 21, 18, 63, 4, 54, 83, 98, 74, 92]
[28, 21, 18, 63, 4, 54, 83, 98, 74, 92, 21]
[21, 18, 63, 4, 54, 83, 98, 74, 92, 21, 52]
[18, 63, 4, 54, 83, 98, 74, 92, 21, 52, 94]
[63, 4, 54, 83, 98, 74, 92, 21, 52, 94, 96]
[4, 54, 83, 98, 74, 92, 21, 52, 94, 96, 80]
[54, 83, 98, 74, 92, 21, 52, 94, 96, 80, 18]
[83, 98, 74, 92, 21, 52, 94, 96, 80, 18, 65]
[98, 74, 92, 21, 52, 94, 96, 80, 18, 65, 96]
[74, 92, 21, 52, 94, 96, 80, 18, 65, 96, 48]
[92, 21, 52, 94, 96, 80, 18, 65, 96, 48, 79]
[21, 52, 94, 96, 80, 18, 65, 96, 48, 79, 34]
[52, 94, 96, 80, 18, 65, 96, 48, 79, 34, 72]
[94, 96, 80, 18, 65, 96, 48, 79, 34, 72, 95]
[96, 80, 18, 65, 96, 48, 79, 34, 72, 95, 89]
[80, 18, 65, 96, 48, 79, 34, 72, 95, 89, 61]
[18, 65, 96, 48, 79, 34, 72, 95, 89, 61, 86]
[65, 96, 48, 79, 34, 72, 95, 89, 61, 86, 44]
[96, 48, 79, 34, 72, 95, 89, 61, 86, 44, 55]
[48, 79, 34, 72, 95, 89, 61, 86, 44, 55, 61]
[79, 34, 72, 95, 89, 61, 86, 44, 55, 61, 45]
[34, 72, 95, 89, 61, 86, 44, 55, 61, 45, 8]
[72, 95, 89, 61, 86, 44, 55, 61, 45, 8, 72]
[95, 89, 61, 86, 44, 55, 61, 45, 8, 72, 81]
[89, 61, 86, 44, 55, 61, 45, 8, 72, 81, 69]
[61, 86, 44, 55, 61, 45, 8, 72, 81, 69, 26]
[86, 44, 55, 61, 45, 8, 72, 81, 69, 26, 4]
[44, 55, 61, 45, 8, 72, 81, 69, 26, 4, 79]
[55, 61, 45, 8, 72, 81, 69, 26, 4, 79, 86]
[61, 45, 8, 72, 81, 69, 26, 4, 79, 86, 7]
[45, 8, 72, 81, 69, 26, 4, 79, 86, 7, 81]
[8, 72, 81, 69, 26, 4, 79, 86, 7, 81, 66]
[72, 81, 69, 26, 4, 79, 86, 7, 81, 66, 25]
[81, 69, 26, 4, 79, 86, 7, 81, 66, 25, 35]
[69, 26, 4, 79, 86, 7, 81, 66, 25, 35, 9]
[26, 4, 79, 86, 7, 81, 66, 25, 35, 9, 24]
[4, 79, 86, 7, 81, 66, 25, 35, 9, 24, 92]
---------------------------------------------------------------------------
KeyboardInterrupt                         Traceback (most recent call last)
<ipython-input-1-4c3bc8db8184> in <module>()
     90     consumer = connect_kafka_consumer()
     91     fig, ax = init_plots()
---> 92     consume_messages(consumer, fig, ax)
     93
```

```
     94

<ipython-input-1-4c3bc8db8184> in consume_messages(consumer, fig, ax)
     79                 annotate_max(x,y)
     80                 annotate_min(x,y)
---> 81                 fig.canvas.draw()
     82                 x.pop(0) # removing the item in the first position
     83                 y.pop(0)

~/.local/lib/python3.5/site-packages/matplotlib/backends/backend_webagg_core.py in
draw(self)
    148         self._png_is_old = True
    149         try:
--> 150             super().draw()
    151         finally:
    152             self.manager.refresh_all()  # Swap the frames.

~/.local/lib/python3.5/site-packages/matplotlib/backends/backend_agg.py in draw(self)
    400         toolbar = self.toolbar
    401         try:
--> 402             self.figure.draw(self.renderer)
    403             # A GUI class may be need to update a window using this draw, so
    404             # don't forget to call the superclass.

~/.local/lib/python3.5/site-packages/matplotlib/artist.py in draw_wrapper(artist,
renderer, *args, **kwargs)
     48                 renderer.start_filter()
     49
---> 50             return draw(artist, renderer, *args, **kwargs)
     51         finally:
     52             if artist.get_agg_filter() is not None:

~/.local/lib/python3.5/site-packages/matplotlib/figure.py in draw(self, renderer)
   1647
   1648             mimage._draw_list_compositing_images(
-> 1649                 renderer, self, artists, self.suppressComposite)
   1650
   1651             renderer.close_group('figure')

~/.local/lib/python3.5/site-packages/matplotlib/image.py in
_draw_list_compositing_images(renderer, parent, artists, suppress_composite)
    136     if not_composite or not has_images:
    137         for a in artists:
--> 138             a.draw(renderer)
    139     else:
    140         # Composite any adjacent images together

~/.local/lib/python3.5/site-packages/matplotlib/artist.py in draw_wrapper(artist,
renderer, *args, **kwargs)
     48                 renderer.start_filter()
     49
---> 50             return draw(artist, renderer, *args, **kwargs)
     51         finally:
     52             if artist.get_agg_filter() is not None:
```

```
~/.local/lib/python3.5/site-packages/matplotlib/axes/_base.py in draw(self, renderer,
inframe)
   2626             renderer.stop_rasterizing()
   2627
-> 2628         mimage._draw_list_compositing_images(renderer, self, artists)
   2629
   2630         renderer.close_group('axes')

~/.local/lib/python3.5/site-packages/matplotlib/image.py in
_draw_list_compositing_images(renderer, parent, artists, suppress_composite)
    136     if not_composite or not has_images:
    137         for a in artists:
--> 138             a.draw(renderer)
    139     else:
    140         # Composite any adjacent images together

~/.local/lib/python3.5/site-packages/matplotlib/artist.py in draw_wrapper(artist,
renderer, *args, **kwargs)
     48                 renderer.start_filter()
     49
---> 50             return draw(artist, renderer, *args, **kwargs)
     51         finally:
     52             if artist.get_agg_filter() is not None:

~/.local/lib/python3.5/site-packages/matplotlib/axis.py in draw(self, renderer,
*args, **kwargs)
   1185         ticks_to_draw = self._update_ticks(renderer)
   1186         ticklabelBoxes, ticklabelBoxes2 =
self._get_tick_bboxes(ticks_to_draw,
-> 1187                                               renderer)
   1188
   1189         for tick in ticks_to_draw:

~/.local/lib/python3.5/site-packages/matplotlib/axis.py in _get_tick_bboxes(self,
ticks, renderer)
   1123         for tick in ticks:
   1124             if tick.label1On and tick.label1.get_visible():
-> 1125                 extent = tick.label1.get_window_extent(renderer)
   1126                 ticklabelBoxes.append(extent)
   1127             if tick.label2On and tick.label2.get_visible():

~/.local/lib/python3.5/site-packages/matplotlib/text.py in get_window_extent(self,
renderer, dpi)
    928             raise RuntimeError('Cannot get window extent w/o renderer')
    929
--> 930         bbox, info, descent = self._get_layout(self._renderer)
    931         x, y = self.get_unitless_position()
    932         x, y = self.get_transform().transform_point((x, y))

~/.local/lib/python3.5/site-packages/matplotlib/text.py in _get_layout(self,
renderer)
    337
    338         # get the rotation matrix
--> 339         M = Affine2D().rotate_deg(self.get_rotation())
    340
```

```
    341         offsetLayout = np.zeros((len(lines), 2))

~/.local/lib/python3.5/site-packages/matplotlib/transforms.py in rotate_deg(self,
degrees)
   1987         and :meth:`scale`.
   1988         """
-> 1989         return self.rotate(np.deg2rad(degrees))
   1990
   1991     def rotate_around(self, x, y, theta):

~/.local/lib/python3.5/site-packages/matplotlib/transforms.py in rotate(self, theta)
   1976                                 float)
   1977         self._mtx = np.dot(rotate_mtx, self._mtx)
-> 1978         self.invalidate()
   1979         return self
   1980

~/.local/lib/python3.5/site-packages/matplotlib/transforms.py in invalidate(self)
    130         if self.is_affine:
    131             value = self.INVALID_AFFINE
--> 132         return self._invalidate_internal(value, invalidating_node=self)
    133
    134     def _invalidate_internal(self, value, invalidating_node):

~/.local/lib/python3.5/site-packages/matplotlib/transforms.py in
_invalidate_internal(self, value, invalidating_node)
    150
    151         if self.pass_through or status_changed:
--> 152             self._invalid = value
    153
    154             for parent in list(self._parents.values()):

KeyboardInterrupt:
```

In [ ]:

In [ ]: