->Weather<-

Import Libraries

In [1]:
```
import pandas as pd
import numpy as np
import datetime
import matplotlib.pyplot as plt
import seaborn as sns
```

Read File

In [2]:
```
# Assign spreadsheet filename to `file`
file = 'WEATHERDATA.xlsx'
# Load spreadsheet
xl = pd.ExcelFile(file)

# Print the sheet names
print(xl.sheet_names)

# Load a sheet into a DataFrame by name: df1
df = xl.parse('Sheet1')
```

```
['Sheet1']
```

Data Clean, Index, and Fix column names

1. The first three rows don't contain data
2. The last three rows don't contain data
3. Make Date column in index
4. Fix the name of columns

In [3]:
```
df.drop(df.head(3).index,inplace=True)                # Drop the first three rows
df=df[:-3]                                            # Drop the last three rows
df = df.set_index(['Date'])                           # Transform Date column in index
df.index.name = None
df.columns = df.columns.str.replace(' ', '_')         # Fix Column name: Fill spaces with _
df['Value'] = pd.to_numeric(df.Value, errors='coerce')  # Convert Value column to float
```

In [4]:
```
df.head()
```

Out[4]:

	Tag_Name	Value	Value_Type	Circuit_Type	Consumption	Circuit	Facility	Distrito	Contr
2018-01-01 00:00:00	Humidity	95.00000	Measured	Conforto	0.0	Temperatura Local	Arnado	Coimbra	PT0002000068373747
2018-01-01 00:00:00	Temperature	7.85000	Measured	Conforto	0.0	Temperatura Local	Arnado	Coimbra	PT0002000068373747
2018-01-01 01:00:00	Humidity	94.66667	Measured	Conforto	0.0	Temperatura Local	Arnado	Coimbra	PT0002000068373747
2018-01-01 01:00:00	Temperature	7.63333	Measured	Conforto	0.0	Temperatura Local	Arnado	Coimbra	PT0002000068373747
2018-01-01 02:00:00	Humidity	95.00000	Measured	Conforto	0.0	Temperatura Local	Arnado	Coimbra	PT0002000068373747

Filter out Temperature and Humidity

In [5]:
```
temperature = df["Tag_Name"] == "Temperature"
temperature = df[temperature]

humidity = df["Tag_Name"] == "Humidity"
humidity = df[humidity]
```

In [6]:
```
humidity.describe()
```

Out[6]:

	Value	Consumption	Contracted_Power
count	8617.000000	8617.0	8617.0
mean	77.001277	0.0	71.0
std	16.761831	0.0	0.0
min	13.500000	0.0	71.0
25%	66.750000	0.0	71.0
50%	82.500000	0.0	71.0
75%	90.500000	0.0	71.0
max	100.000000	0.0	71.0

In [7]:
```
temperature.describe()
```

Out[7]:

	Value	Consumption	Contracted_Power
count	8617.000000	8617.0	8617.0
mean	15.849543	0.0	71.0
std	6.128860	0.0	0.0
min	-0.350000	0.0	71.0
25%	11.525000	0.0	71.0
50%	15.200000	0.0	71.0
75%	19.300000	0.0	71.0
max	41.000000	0.0	71.0

Verify Nulls and Missing values

In [24]:
```
#
humidity.isnull().values.any()  #Are there null values?
humidity.isna().values.any()    #Are there missing values?
```

Out[24]:
```
False
```

In [25]:
```
temperature.isnull().values.any()  #Are there null values?
temperature.isna().values.any()    #Are there missing values?
```

Out[25]:
```
False
```

Hold on only Important Columns

1. Verify unique values
2. Delete columns without unique values

Humidity

In [26]:
```
humidity.nunique()
```

Out[26]:
```
Tag_Name                   1
Value                    356
Value_Type                 1
Circuit_Type               1
Consumption                1
Circuit                    1
Facility                   1
Distrito                   1
Contract                   1
Destination_Name           1
Destination_Email          1
Tariff_Type                1
Municipality               1
Contracted_Power           1
Solar_orientation          1
Summer_climatic_zone       1
Winter_climatic_zone       1
dtype: int64
```

In [27]:
```
hum_small=humidity['Value']
hum= pd.DataFrame(hum_small)                    #converte Series to DataFrame
hum=hum.rename(columns={'Value':'Humidity'})  #rename column
```

Temperature

In [28]:
```
temperature.nunique()
```

Out[28]:
```
Tag_Name                    1
Value                    1288
Value_Type                  1
Circuit_Type                1
Consumption                 1
Circuit                     1
Facility                    1
Distrito                    1
Contract                    1
Destination_Name            1
Destination_Email           1
Tariff_Type                 1
Municipality                1
Contracted_Power            1
Solar_orientation           1
Summer_climatic_zone        1
Winter_climatic_zone        1
dtype: int64
```

In [29]:
```
temp_small=temperature['Value']
temp= pd.DataFrame(temp_small)                  #converte Series to DataFrame
temp=temp.rename(columns={'Value':'Temperature'})  #rename column
```

General Statistics

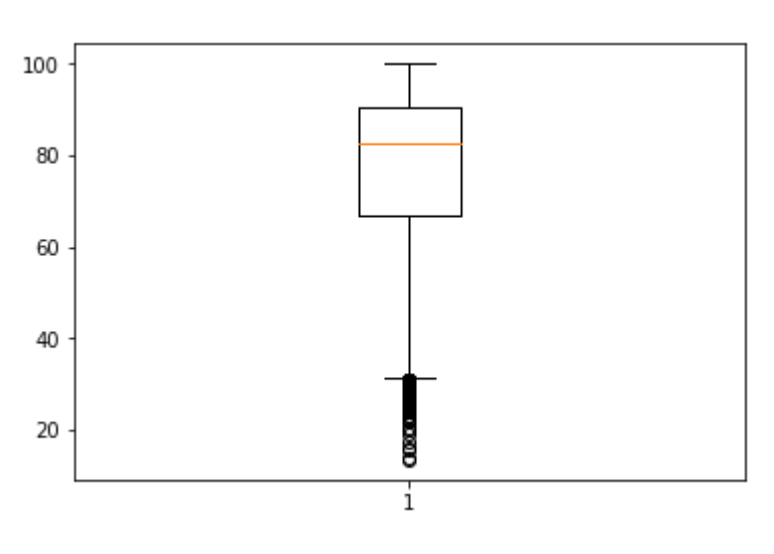
In [30]:
```
hum.describe()
```

Out[30]:

	Humidity
count	8617.000000
mean	77.001277
std	16.761831
min	13.500000
25%	66.750000
50%	82.500000
75%	90.500000
max	100.000000

In [31]:
```
data= hum['Humidity']
plt.boxplot(data)
```

Out[31]:
```
{'whiskers': [<matplotlib.lines.Line2D at 0x1a4d1b11908>,
  <matplotlib.lines.Line2D at 0x1a4d1b11c50>],
 'caps': [<matplotlib.lines.Line2D at 0x1a4d1b11f98>,
  <matplotlib.lines.Line2D at 0x1a4d1ba4320>],
 'boxes': [<matplotlib.lines.Line2D at 0x1a4d1b114e0>],
 'medians': [<matplotlib.lines.Line2D at 0x1a4d1ba4668>],
 'fliers': [<matplotlib.lines.Line2D at 0x1a4d1ba49b0>],
 'means': []}
```



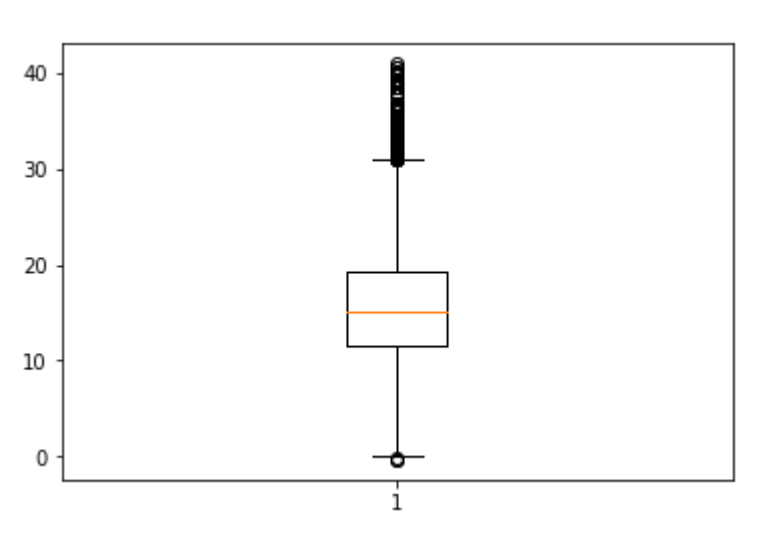
In [32]:
```
temp.describe()
```

Out[32]:

	Temperature
count	8617.000000
mean	15.849543
std	6.128860
min	-0.350000
25%	11.525000
50%	15.200000
75%	19.300000
max	41.000000

In [33]:
```
data= temp['Temperature']
plt.boxplot(data)
```

Out[33]:
```
{'whiskers': [<matplotlib.lines.Line2D at 0x1a4d1e83358>,
  <matplotlib.lines.Line2D at 0x1a4d1e836a0>],
 'caps': [<matplotlib.lines.Line2D at 0x1a4d1e839e8>,
  <matplotlib.lines.Line2D at 0x1a4d1cd6898>],
 'boxes': [<matplotlib.lines.Line2D at 0x1a4d1ea0ef0>],
 'medians': [<matplotlib.lines.Line2D at 0x1a4d1e83e10>],
 'fliers': [<matplotlib.lines.Line2D at 0x1a4d1f73400>],
 'means': []}
```

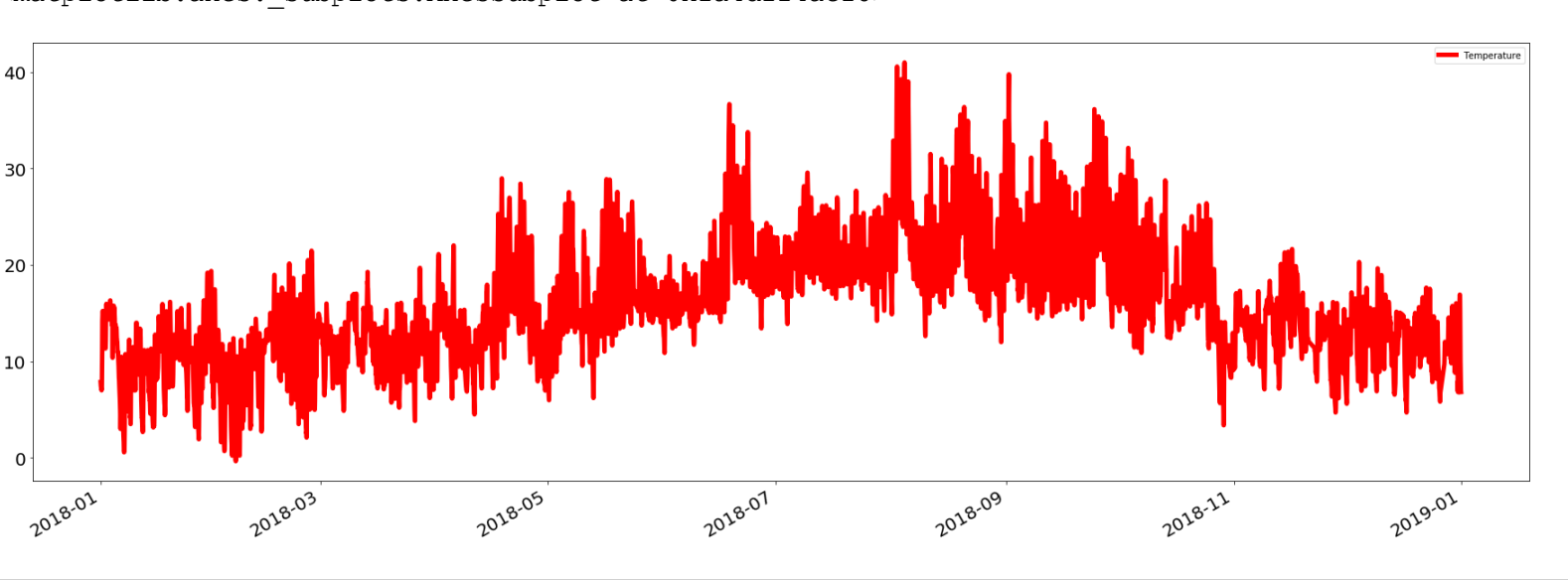


Some Insights

1. The high temperature values are related with summer
2. The low humidity values are related with summer

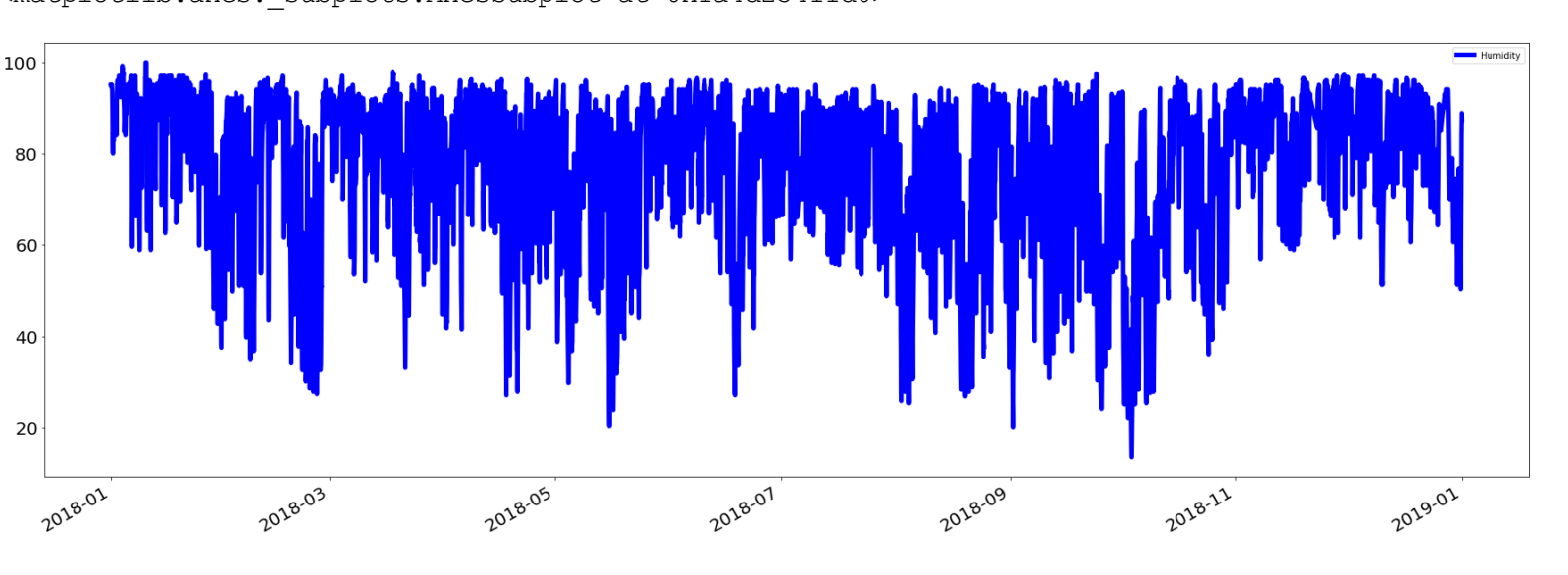
In [34]:
```
# Plot humidity and temperature in one graph
temp.plot(figsize=(30,10), linewidth=5, fontsize=20, color='red')
```

Out[34]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x1a4d1f4acf8>
```

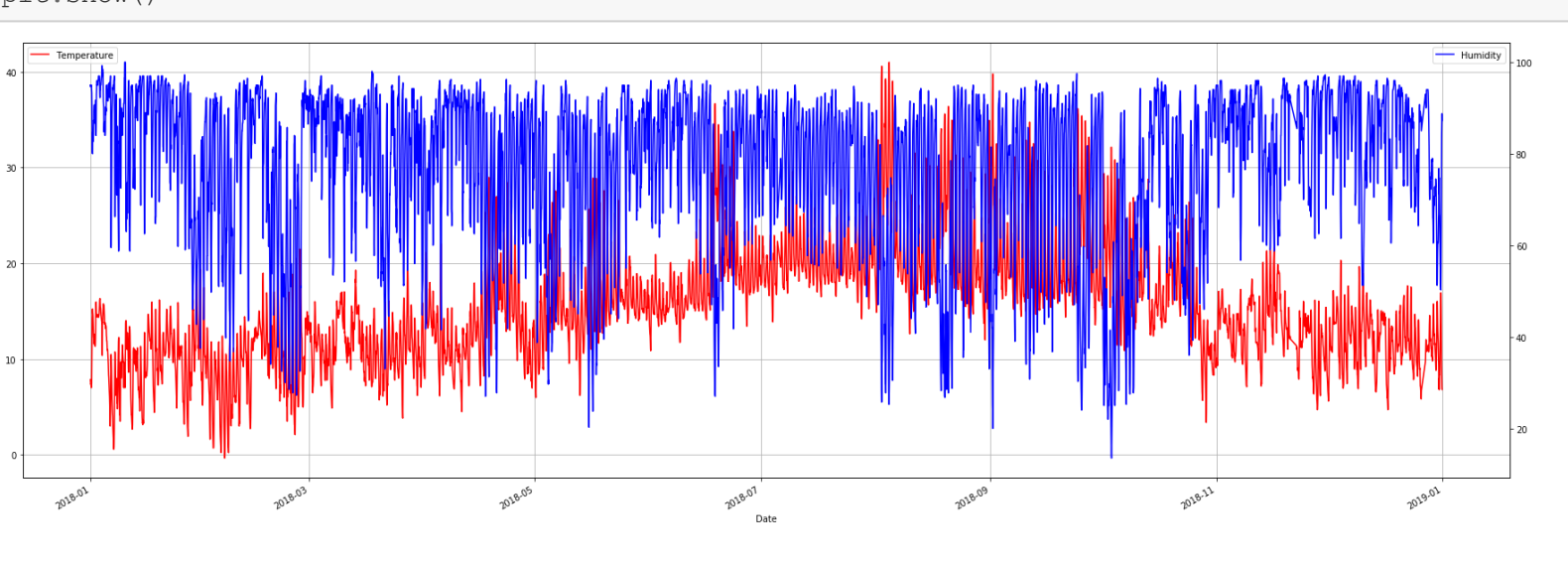


In [35]:
```
hum.plot(figsize=(30,10), linewidth=5, fontsize=20, color='blue')
```

Out[35]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x1a4d23411d0>
```

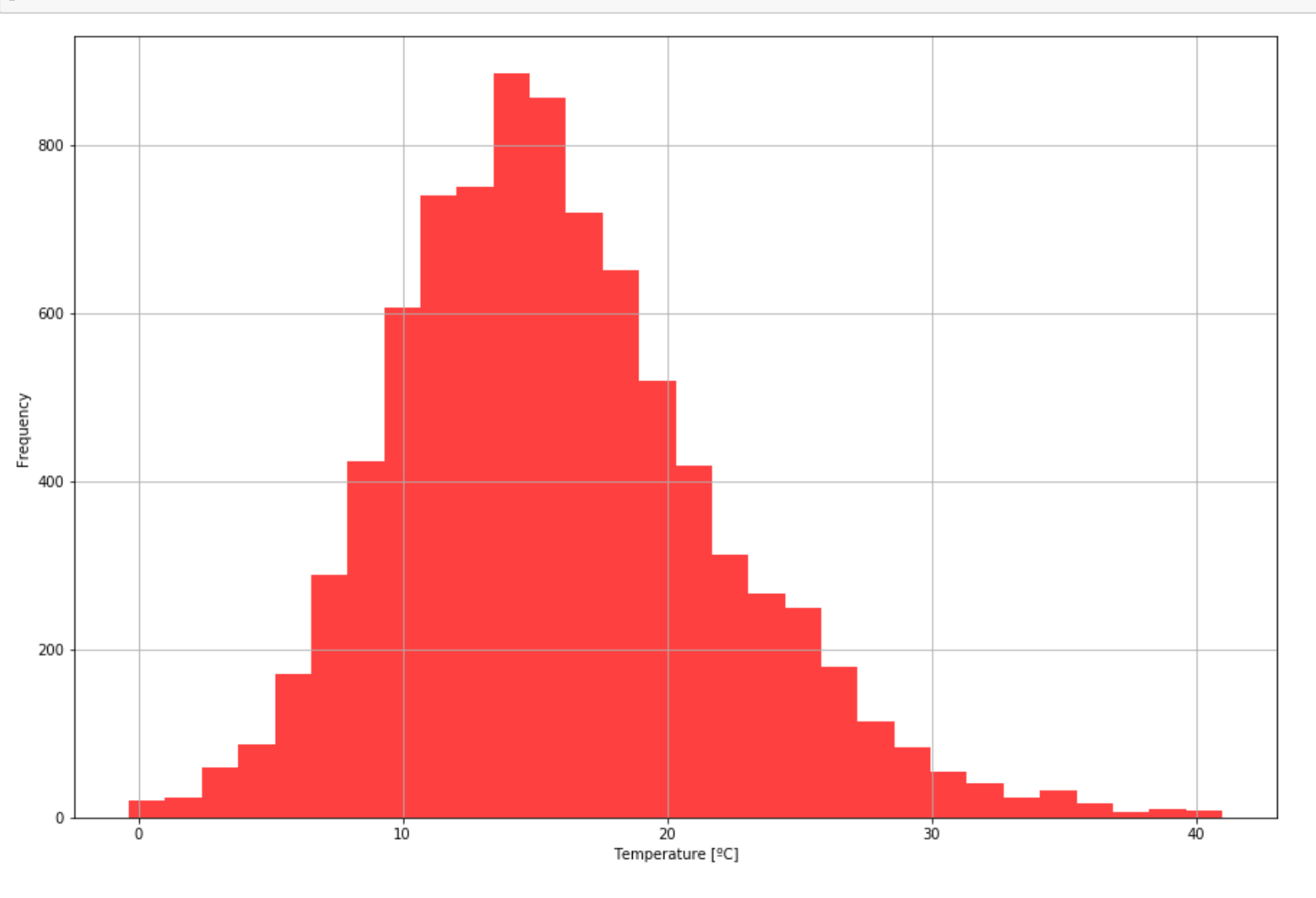


In [36]:
```
plt.figure(figsize=(30,10))
plt.xlabel('Date')

ax1 = temp.Temperature.plot(color='red', grid=True, label='Temperature')
ax2 = hum.Humidity.plot(color='blue', grid=True, secondary_y=True, label='Humidity')

#h1, l1 = ax1.get_legend_handles_labels()
#h2, l2 = ax2.get_legend_handles_labels()
#plt.legend(h1+h2, l1+l2, loc=2)
ax1.legend(loc=2)
ax2.legend(loc=1)


plt.show()
```

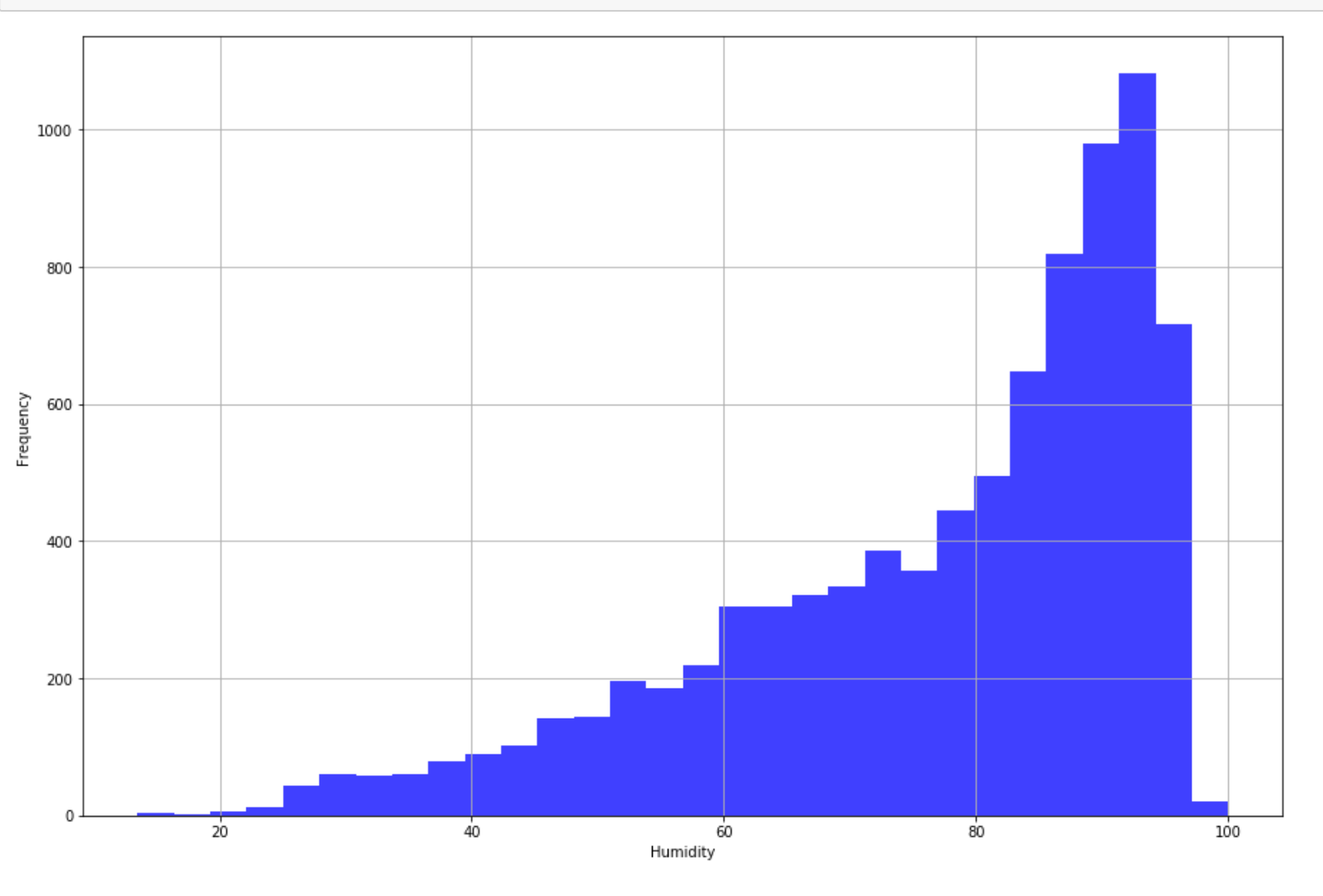


Histograms

In [37]:
```
plt.figure(figsize=(15,10))
x=temp.Temperature
# the histogram of the data
n, bins, patches = plt.hist(x, 30, density=False, facecolor='r', alpha=0.75)
plt.xlabel('Temperature [ºC]')
plt.ylabel('Frequency')
plt.grid(True)
plt.show()
```



In [38]:
```
plt.figure(figsize=(15,10))
x=hum.Humidity
# the histogram of the data
n, bins, patches = plt.hist(x, 30, density=False, facecolor='b', alpha=0.75)
plt.xlabel('Humidity')
plt.ylabel('Frequency')
plt.grid(True)
plt.show()
```

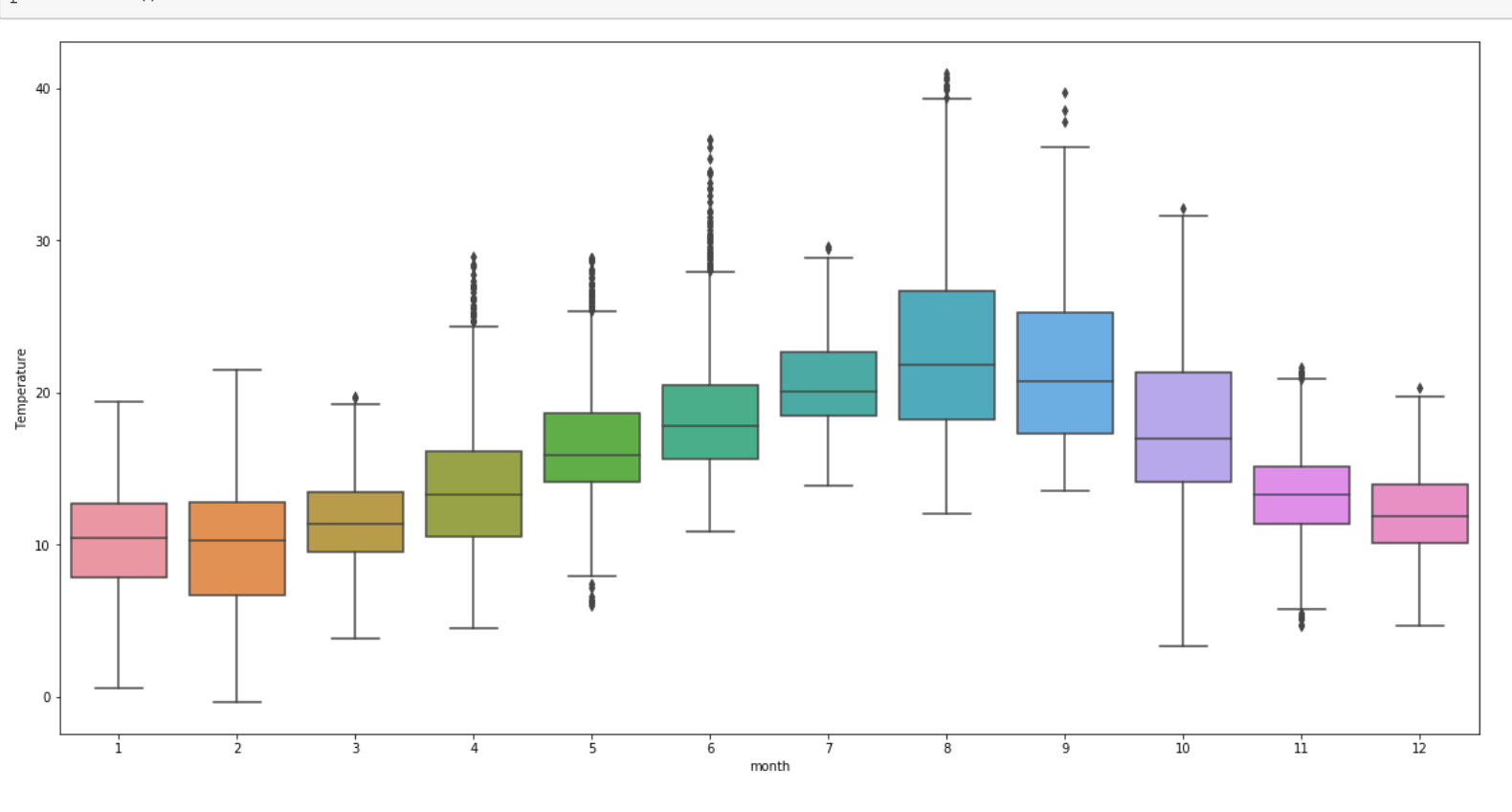


Boxplot per Month

Temperature

In [39]:
```
temp['year'] = temp.index.year
temp['month'] = temp.index.month
temp['day'] = temp.index.day
```

In [40]:
```
fig, ax = plt.subplots()
fig.set_size_inches((20,10))
sns.boxplot(x='month',y='Temperature',data=temp,ax=ax)
plt.show()
```



In [65]:
```
grouped = temp.groupby('month').agg({"Temperature":[min,max]})
grouped.columns = ["_".join(x) for x in grouped.columns.ravel()]
grouped['mean']=temp.groupby('month')['Temperature'].mean()
grouped['median']=temp.groupby('month')['Temperature'].median()
grouped['std']=temp.groupby('month')['Temperature'].std()
grouped
```

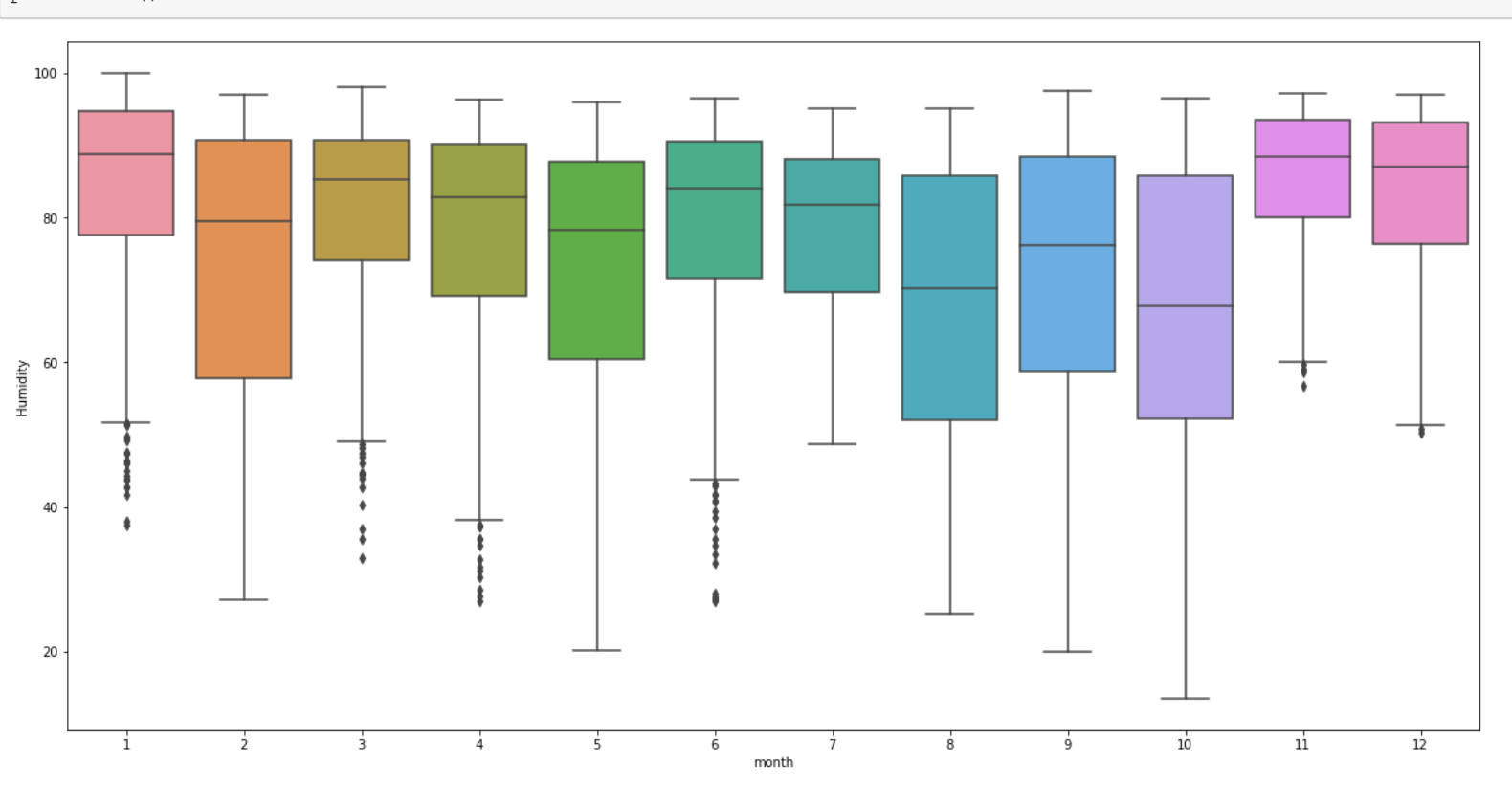
Out[65]:

	Temperature_min	Temperature_max	mean	median	std
month					
1	0.57500	19.37500	10.229955	10.4250	3.593552
2	-0.35000	21.47500	9.899434	10.2500	4.377770
3	3.82500	19.70000	11.449536	11.3750	2.838028
4	4.50000	28.97500	14.004167	13.3125	4.655026
5	5.97500	28.90000	16.655866	15.8500	4.227379
6	10.87500	36.67500	18.928077	17.7750	4.578857
7	13.87500	29.57500	20.690595	20.0750	2.922618
8	12.00000	41.00000	23.064975	21.8250	5.979016
9	13.56667	39.77500	21.754829	20.7250	5.150778
10	3.37500	32.13333	17.647695	16.9750	5.094984
11	4.70000	21.65000	13.239858	13.2875	3.103577
12	4.70000	20.30000	11.974281	11.9000	2.826487

Humidity

In [41]:
```
hum['year'] = hum.index.year
hum['month'] = hum.index.month
hum['day'] = hum.index.day
```

In [42]:
```
fig, ax = plt.subplots()
fig.set_size_inches((20,10))
sns.boxplot(x='month',y='Humidity',data=hum,ax=ax)
plt.show()
```



In [70]:
```
grouped = hum.groupby('month').agg({'Humidity':[min,max]})
grouped.columns = ["_".join(x) for x in grouped.columns.ravel()]
grouped['mean']=hum.groupby('month')['Humidity'].mean()
grouped['median']=hum.groupby('month')['Humidity'].median()
grouped['std']=hum.groupby('month')['Humidity'].std()
grouped
```

Out[70]:

	Humidity_min	Humidity_max	mean	median	std
month					
1	37.50	100.00	84.403491	88.750	12.923458
2	27.25	97.00	73.484037	79.500	19.420575
3	33.00	98.00	80.875906	85.250	12.816816
4	27.00	96.25	78.348095	82.875	15.053796
5	20.25	96.00	73.267048	78.250	17.592041
6	27.00	96.50	79.598834	84.000	14.143816
7	48.75	95.00	78.582757	81.750	11.158331
8	25.25	95.00	68.116487	70.250	19.422288
9	20.00	97.50	71.864627	76.250	18.659885
10	13.50	96.50	66.835807	67.750	20.542315
11	56.75	97.25	85.376344	88.500	9.839231
12	50.25	97.00	83.839824	87.000	10.998461