Data science exploration

Featuring @FremdBot temperature data

- Reminder 1: What is a data scientist?
  - Video: https://www.youtube.com/watch?v=i2jwZcWicSY (https://www.youtube.com/watch?v=i2jwZcWicSY)

- Reminder 2: What is Pandas?
  - Pandas is a popular tool that data scientists use to analyze data sets
  - It is a library designed to make it easier to work with large datasets
  - Pandas offers users 2 new data structures:
    - series
    - dataframes

- Background for today's data exploration:
  - @FremdBot is a Twitter account that tweets out temperature data from Fremd https://twitter.com/fremdbot (https://twitter.com/fremdbot)
  - Made from two Raspberry Pi computers, two temperature probes, and a camera
  - The data collected by @FremdBot is publicly available here:
    - https://thingspeak.com/channels/123549 (https://thingspeak.com/channels/123549)
    - https://thingspeak.com/channels/142171 (https://thingspeak.com/channels/142171)
    - This is an example of "open data" (data that is freely shared with the world)
    - Further reading: https://en.wikipedia.org/wiki/Open_data (https://en.wikipedia.org/wiki/Open_data)
      - Related terms:
      - The Internet of Things (https://en.wikipedia.org/wiki/Internet_of_things (https://en.wikipedia.org/wiki/Internet_of_things))
      - Citizen Science (https://en.wikipedia.org/wiki/Citizen_science (https://en.wikipedia.org/wiki/Citizen_science))

The @FremdBot project has generated a great deal of data

- We will use Pandas and some of the tools of datascience to dive into this data
- The data structure we will be using is a dataframe:

In [27]:
```
# Use pandas to read in the temperature data
import pandas as pd

# Reads in a csv file (comma separated values)
original_data = pd.read_csv("Outdoor_2019.csv")

# Show the last 3 entries in the file:
original_data[-3: ]
```

Out[27]:

	date_time	temp
24940	2017-09-16 17:05:42 UTC	87.6866
24941	2017-09-16 18:05:41 UTC	86.7866
24942	2017-09-16 19:05:42 UTC	87.3500

Clean the data (remove erroneous temperatures)

- Cleaning data is the process of finding and deleting/fixing erroneous data
  - Further reading: https://en.wikipedia.org/wiki/Data_cleansing (https://en.wikipedia.org/wiki/Data_cleansing)
  - In this example, the Raspberry Pi sometimes records an erroneous reading of 185 degrees F

In [28]:
```
# Print length
print(len(original_data))

# Trim bad data
temp_data = original_data[original_data.temp < 120]

# Print new length
print(len(temp_data))
```

```
24943
24935
```

Look at all the temperatures (and sort from low to high)

In [29]:
```
# Sort the pandas dataframe by temperature
temp_data = temp_data.sort_values("temp")

# Print the first 10 items
temp_data[:10]
```

Out[29]:

	date_time	temp
21283	2016-12-19 10:54:46 UTC	-9.2866
21278	2016-12-19 10:29:44 UTC	-9.2866
21279	2016-12-19 10:34:44 UTC	-9.2866
21292	2016-12-19 11:39:51 UTC	-9.2866
21293	2016-12-19 11:44:51 UTC	-9.2866
21280	2016-12-19 10:39:45 UTC	-9.2866
21275	2016-12-19 10:14:43 UTC	-9.1750
21276	2016-12-19 10:19:43 UTC	-9.1750
21277	2016-12-19 10:24:43 UTC	-9.1750
21281	2016-12-19 10:44:45 UTC	-9.1750

Now create a single list (pandas series) of only temperatures

In [30]:
```
# Create variable, store "temp" values
temps_only = temp_data["temp"]

# Print first ten items
temps_only[:10]
```

Out[30]:
```
21283   -9.2866
21278   -9.2866
21279   -9.2866
21292   -9.2866
21293   -9.2866
21280   -9.2866
21275   -9.1750
21276   -9.1750
21277   -9.1750
21281   -9.1750
Name: temp, dtype: float64
```

Find the average temperature

- The average annual temperature in the Chicagoland area is usually between 49.5F and 52.5F, so this number seems reasonable:

In [31]:
```
# Import numpy
import numpy as np

# Create list/array of temperatures
temp_array = np.array(temps_only)

# Calculate and print the average
avg = np.average(temp_array)
avg
```

Out[31]:
```
50.88674208141167
```

Find all temperatures greater than 50

In [32]:
```
# Create variable, store only temps above 50
above_50 = temp_data[temps_only > 50]

# Print the first ten values
above_50
```

Out[32]:

	date_time	temp
17254	2016-11-29 02:25:29 UTC	50.1116
13614	2016-11-11 17:12:36 UTC	50.1116
11880	2016-11-03 14:16:21 UTC	50.1116
4099	2016-09-28 12:29:15 UTC	50.1116
14921	2016-11-16 23:12:14 UTC	50.1116
17257	2016-11-29 02:40:31 UTC	50.1116
17256	2016-11-29 02:35:30 UTC	50.1116
17255	2016-11-29 02:30:30 UTC	50.1116
24405	2017-08-25 10:05:42 UTC	50.1116
4101	2016-09-28 12:39:16 UTC	50.1116
13126	2016-11-09 02:33:19 UTC	50.1116
9484	2016-10-21 20:49:10 UTC	50.1116
8757	2016-10-19 08:07:47 UTC	50.1116
6950	2016-10-09 08:23:37 UTC	50.1116
13894	2016-11-12 16:34:36 UTC	50.1116
13594	2016-11-11 15:32:28 UTC	50.1116
17290	2016-11-29 05:25:45 UTC	50.1116
4103	2016-09-28 12:49:17 UTC	50.1116
9493	2016-10-21 21:34:14 UTC	50.1116
13864	2016-11-12 14:04:24 UTC	50.1116
14636	2016-11-15 23:25:01 UTC	50.1116
6953	2016-10-09 08:38:38 UTC	50.1116
13374	2016-11-09 23:15:14 UTC	50.1116
14268	2016-11-13 23:47:27 UTC	50.1116
6902	2016-10-09 04:23:18 UTC	50.1116
13549	2016-11-10 13:51:28 UTC	50.1116
17251	2016-11-29 02:10:28 UTC	50.1116
17211	2016-11-28 22:50:06 UTC	50.1116
17227	2016-11-29 00:10:18 UTC	50.1116
17228	2016-11-29 00:15:18 UTC	50.1116
...	...	...
23817	2017-07-31 20:05:41 UTC	89.9366

	date_time	temp
23530	2017-07-07 00:05:41 UTC	89.9366
23839	2017-08-01 18:05:41 UTC	90.0500
37	2016-09-07 20:25:43 UTC	90.1616
41	2016-09-07 20:45:45 UTC	90.2750
45	2016-09-07 21:05:47 UTC	90.2750
38	2016-09-07 20:30:44 UTC	90.3866
39	2016-09-07 20:35:44 UTC	90.3866
40	2016-09-07 20:40:45 UTC	90.3866
23627	2017-07-23 21:05:41 UTC	90.5000
23864	2017-08-02 19:05:42 UTC	90.5000
23865	2017-08-02 20:05:41 UTC	90.6116
23697	2017-07-26 19:05:43 UTC	90.6116
23502	2017-07-05 20:05:41 UTC	90.7250
24295	2017-08-20 18:05:41 UTC	90.8366
23840	2017-08-01 19:05:42 UTC	90.8366
23503	2017-07-05 21:05:41 UTC	90.9500
23623	2017-07-23 17:05:42 UTC	91.1750
23626	2017-07-23 20:05:41 UTC	91.2866
23529	2017-07-06 23:05:41 UTC	91.5116
23816	2017-07-31 19:05:41 UTC	91.5116
23625	2017-07-23 19:05:41 UTC	91.6250
23862	2017-08-02 17:05:41 UTC	91.9616
23528	2017-07-06 22:05:42 UTC	92.8616
23624	2017-07-23 18:05:42 UTC	93.3116
23523	2017-07-06 17:05:41 UTC	93.3116
23524	2017-07-06 18:05:42 UTC	93.5366
23527	2017-07-06 21:05:42 UTC	94.1000
23525	2017-07-06 19:05:42 UTC	95.0000
23526	2017-07-06 20:05:42 UTC	95.4500

13483 rows × 2 columns

Find all temperatures greater than 90

In [33]:
```
above_90 = temp_data[temps_only > 90]

# Print the first ten values
above_90[:10]
```

Out[33]:

	date_time	temp
23839	2017-08-01 18:05:41 UTC	90.0500
37	2016-09-07 20:25:43 UTC	90.1616
41	2016-09-07 20:45:45 UTC	90.2750
45	2016-09-07 21:05:47 UTC	90.2750
38	2016-09-07 20:30:44 UTC	90.3866
39	2016-09-07 20:35:44 UTC	90.3866
40	2016-09-07 20:40:45 UTC	90.3866
23627	2017-07-23 21:05:41 UTC	90.5000
23864	2017-08-02 19:05:42 UTC	90.5000
23865	2017-08-02 20:05:41 UTC	90.6116

Look at the 10 highest temperatures

In [34]:
```
# Create variable, store the last 10 temps
high_temps = above_90[-10:]

# Print the first ten values
high_temps
```

Out[34]:

	date_time	temp
23816	2017-07-31 19:05:41 UTC	91.5116
23625	2017-07-23 19:05:41 UTC	91.6250
23862	2017-08-02 17:05:41 UTC	91.9616
23528	2017-07-06 22:05:42 UTC	92.8616
23624	2017-07-23 18:05:42 UTC	93.3116
23523	2017-07-06 17:05:41 UTC	93.3116
23524	2017-07-06 18:05:42 UTC	93.5366
23527	2017-07-06 21:05:42 UTC	94.1000
23525	2017-07-06 19:05:42 UTC	95.0000
23526	2017-07-06 20:05:42 UTC	95.4500

When did those high temperatures occur?

In [35]:
```
# Create a variable, store only dates
high_temp_dates = high_temps["date_time"]

# Print results
high_temp_dates
```

Out[35]:
```
23816    2017-07-31 19:05:41 UTC
23625    2017-07-23 19:05:41 UTC
23862    2017-08-02 17:05:41 UTC
23528    2017-07-06 22:05:42 UTC
23624    2017-07-23 18:05:42 UTC
23523    2017-07-06 17:05:41 UTC
23524    2017-07-06 18:05:42 UTC
23527    2017-07-06 21:05:42 UTC
23525    2017-07-06 19:05:42 UTC
23526    2017-07-06 20:05:42 UTC
Name: date_time, dtype: object
```

Reformat the date and time of each entry

In [36]:
```
# Import datetime
from datetime import datetime
new_dates = []
for date in temp_data["date_time"]:
    new_dates.append(datetime.strptime(date,'%Y-%m-%d %H:%M:%S UTC'))

print(new_dates[:10])
```

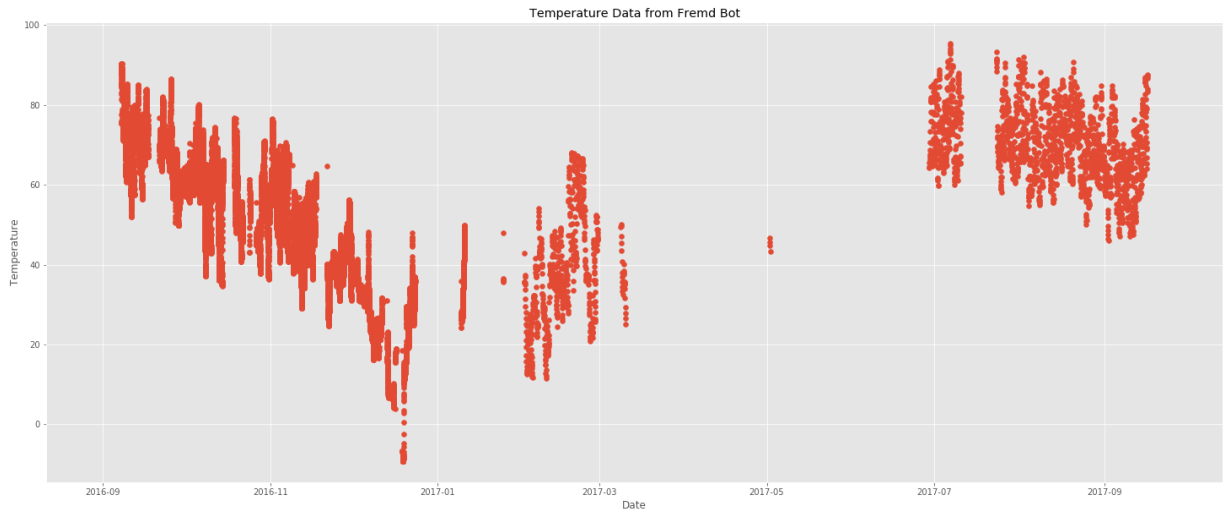
```
[datetime.datetime(2016, 12, 19, 10, 54, 46), datetime.datetime(2016, 12, 19, 1
0, 29, 44), datetime.datetime(2016, 12, 19, 10, 34, 44), datetime.datetime(201
6, 12, 19, 11, 39, 51), datetime.datetime(2016, 12, 19, 11, 44, 51), datetime.d
atetime(2016, 12, 19, 10, 39, 45), datetime.datetime(2016, 12, 19, 10, 14, 43),
datetime.datetime(2016, 12, 19, 10, 19, 43), datetime.datetime(2016, 12, 19, 1
0, 24, 43), datetime.datetime(2016, 12, 19, 10, 44, 45)]
```

Now to graph the temperature data...

In [37]:
```
# Import matplotlib
import matplotlib.pyplot as plot
%matplotlib inline

# Graph details
plot.figure(2,(25,10))
plot.style.use("ggplot")  # fivethirtyeight, bmh, grayscale, dark_background, ggp
plot.title("Temperature Data from Fremd Bot")
plot.xlabel("Date")
plot.ylabel("Temperature")

# plot.scatter(x-axis data, y-axis data)
plot.scatter(new_dates,temp_data["temp"])
plot.show()
```



Observations:

- Sometimes a good visualization can show us something that we don't notice in the full data set
  - What do you notice about this scatterplot that wasn't obvious in the original dataset?
  - Is this dataset complete?
  - What might have happened with the data?
    - Instructor note: Two main things happened to the data. Sometimes the .py script that recorded temperatures would crash and need to be rebooted. Othertimes the power to our raspberry pi device would be cut off :(Both of these scenarios would cause gaps in the data.
  - Should a dateset be disregarded or deleted because it is imperfect?

Try again with indoor temperatures (Room 223):

In [38]:
```
# Import pandas
import pandas as pd

# Read in the csv file (comma separated values)
indoor_orig = pd.read_csv('Room_223_2019.csv')

# Clean the data (remove entries over 120F)
indoor_clean = indoor_orig[indoor_orig.temp<120]

# Reformat the dates and times
from datetime import datetime
indoor_dates = []

for date in indoor_clean["date_time"]:
    indoor_dates.append(datetime.strptime(date,'%Y-%m-%d %H:%M:%S UTC'))

# Print first 10 results
print(indoor_dates[:10])
```

```
[datetime.datetime(2016, 8, 12, 22, 29, 30), datetime.datetime(2016, 8, 12, 22,
34, 31), datetime.datetime(2016, 8, 12, 22, 39, 31), datetime.datetime(2016, 8,
12, 22, 44, 31), datetime.datetime(2016, 8, 12, 22, 49, 32), datetime.datetime
(2016, 8, 12, 22, 54, 32), datetime.datetime(2016, 8, 12, 22, 59, 33), datetim
e.datetime(2016, 8, 12, 23, 4, 33), datetime.datetime(2016, 8, 12, 23, 9, 33),
datetime.datetime(2016, 8, 12, 23, 14, 34)]
```

In [39]:
```
# Use pandas to read in the temperature data
import pandas as pd

# Reads in a csv file (comma separated values)
new_original_data = pd.read_csv("Room_223_2019.csv")

# Show the last 3 entries in the file:
new_original_data[-3: ]
```

Out[39]:

	date_time	temp
130694	2018-01-28 20:21:54 UTC	63.1616
130695	2018-01-28 20:26:55 UTC	63.1616
130696	2018-01-28 20:31:55 UTC	63.1616

In [40]:
```
# Print Length
print(len(new_original_data))

# Trim bad data
temp_data = new_original_data[new_original_data.temp < 120]

# Print new length
print(len(temp_data))
```

```
130697
130687
```

In [41]:
```
temp_data = temp_data.sort_values("temp")

# Print the first 10 items
temp_data[:10]
```

Out[41]:

	date_time	temp
123437	2017-12-27 15:46:28 UTC	54.5000
123436	2017-12-27 15:41:28 UTC	54.5000
123434	2017-12-27 15:31:27 UTC	54.6116
123435	2017-12-27 15:36:27 UTC	54.6116
123431	2017-12-27 15:16:25 UTC	54.6116
123432	2017-12-27 15:21:26 UTC	54.7250
123433	2017-12-27 15:26:26 UTC	54.7250
123429	2017-12-27 15:06:24 UTC	54.7250
123430	2017-12-27 15:11:25 UTC	54.7250
123428	2017-12-27 15:01:24 UTC	54.8366

In [42]:
```
temps_only = temp_data["temp"]

# Print first ten items
temps_only[:10]
```

Out[42]:
```
123437    54.5000
123436    54.5000
123434    54.6116
123435    54.6116
123431    54.6116
123432    54.7250
123433    54.7250
123429    54.7250
123430    54.7250
123428    54.8366
Name: temp, dtype: float64
```

In [43]:
```
below_50 = temp_data[temps_only < 55]

# Print the first ten values
below_50
```

Out[43]:

	date_time	temp
123437	2017-12-27 15:46:28 UTC	54.5000
123436	2017-12-27 15:41:28 UTC	54.5000
123434	2017-12-27 15:31:27 UTC	54.6116
123435	2017-12-27 15:36:27 UTC	54.6116
123431	2017-12-27 15:16:25 UTC	54.6116
123432	2017-12-27 15:21:26 UTC	54.7250
123433	2017-12-27 15:26:26 UTC	54.7250
123429	2017-12-27 15:06:24 UTC	54.7250
123430	2017-12-27 15:11:25 UTC	54.7250
123428	2017-12-27 15:01:24 UTC	54.8366
123427	2017-12-27 14:56:23 UTC	54.9500
123426	2017-12-27 14:51:23 UTC	54.9500

In [44]:
```
above_80 = temp_data[temps_only > 80]

# Print the first ten values
above_80
```

Out[44]:

	date_time	temp
89251	2017-08-02 02:07:44 UTC	80.0366
89250	2017-08-02 02:02:43 UTC	80.0366
89249	2017-08-02 01:57:43 UTC	80.0366
83789	2017-07-12 08:48:26 UTC	80.0366
79830	2017-06-27 00:22:45 UTC	80.0366
982	2016-08-16 08:27:51 UTC	80.0366
981	2016-08-16 08:22:50 UTC	80.0366
980	2016-08-16 08:17:50 UTC	80.0366
979	2016-08-16 08:12:50 UTC	80.0366
83260	2017-07-10 12:41:23 UTC	80.0366
79829	2017-06-27 00:17:44 UTC	80.0366
77061	2017-06-16 18:31:53 UTC	80.0366
78099	2017-06-21 00:07:26 UTC	80.0366
83817	2017-07-12 11:06:12 UTC	80.0366
83791	2017-07-12 08:58:27 UTC	80.0366
83788	2017-07-12 08:43:25 UTC	80.0366
81937	2017-07-04 23:42:37 UTC	80.0366
83787	2017-07-12 08:38:25 UTC	80.0366
89253	2017-08-02 02:17:45 UTC	80.0366
89254	2017-08-02 02:22:45 UTC	80.0366
89255	2017-08-02 02:27:46 UTC	80.0366
89256	2017-08-02 02:32:46 UTC	80.0366
89257	2017-08-02 02:37:47 UTC	80.0366
82513	2017-07-06 23:42:22 UTC	80.0366
81931	2017-07-04 23:12:34 UTC	80.0366
82512	2017-07-06 23:37:22 UTC	80.0366
85725	2017-07-19 02:07:42 UTC	80.0366
78661	2017-06-22 22:57:29 UTC	80.0366
78660	2017-06-22 22:52:29 UTC	80.0366
76149	2017-06-13 14:26:54 UTC	80.0366
...	...	...
75800	2017-06-12 07:33:02 UTC	92.7500

	date_time	temp
75786	2017-06-12 06:22:56 UTC	92.7500
75805	2017-06-12 07:58:06 UTC	92.7500
75804	2017-06-12 07:53:06 UTC	92.7500
75801	2017-06-12 07:38:03 UTC	92.7500
75802	2017-06-12 07:43:04 UTC	92.7500
75789	2017-06-12 06:37:57 UTC	92.7500
75788	2017-06-12 06:32:57 UTC	92.7500
75808	2017-06-12 08:13:09 UTC	92.8616
75843	2017-06-12 11:06:18 UTC	92.8616
75831	2017-06-12 10:06:06 UTC	92.8616
75829	2017-06-12 09:56:06 UTC	92.8616
75827	2017-06-12 09:46:05 UTC	92.8616
75809	2017-06-12 08:18:11 UTC	92.8616
75810	2017-06-12 08:23:11 UTC	92.8616
75814	2017-06-12 08:43:13 UTC	92.8616
75815	2017-06-12 08:48:13 UTC	92.8616
75816	2017-06-12 08:53:14 UTC	92.8616
75817	2017-06-12 08:58:14 UTC	92.8616
75818	2017-06-12 09:00:59 UTC	92.8616
75819	2017-06-12 09:06:00 UTC	92.8616
75820	2017-06-12 09:11:00 UTC	92.8616
75821	2017-06-12 09:16:01 UTC	92.8616
75822	2017-06-12 09:21:01 UTC	92.8616
75823	2017-06-12 09:26:02 UTC	92.8616
75824	2017-06-12 09:31:03 UTC	92.8616
75825	2017-06-12 09:36:03 UTC	92.8616
75826	2017-06-12 09:41:04 UTC	92.8616
75830	2017-06-12 10:01:06 UTC	92.8616
75828	2017-06-12 09:51:05 UTC	92.8616

8897 rows × 2 columns

In [45]:
```
# Create variable, store the last 10 temps
low_temps = below_50[:10]

# Print the first ten values
low_temps
```

Out[45]:

	date_time	temp
123437	2017-12-27 15:46:28 UTC	54.5000
123436	2017-12-27 15:41:28 UTC	54.5000
123434	2017-12-27 15:31:27 UTC	54.6116
123435	2017-12-27 15:36:27 UTC	54.6116
123431	2017-12-27 15:16:25 UTC	54.6116
123432	2017-12-27 15:21:26 UTC	54.7250
123433	2017-12-27 15:26:26 UTC	54.7250
123429	2017-12-27 15:06:24 UTC	54.7250
123430	2017-12-27 15:11:25 UTC	54.7250
123428	2017-12-27 15:01:24 UTC	54.8366

In [46]:
```
high_temps = above_80[-10:]

# Print the first ten values
high_temps
```

Out[46]:

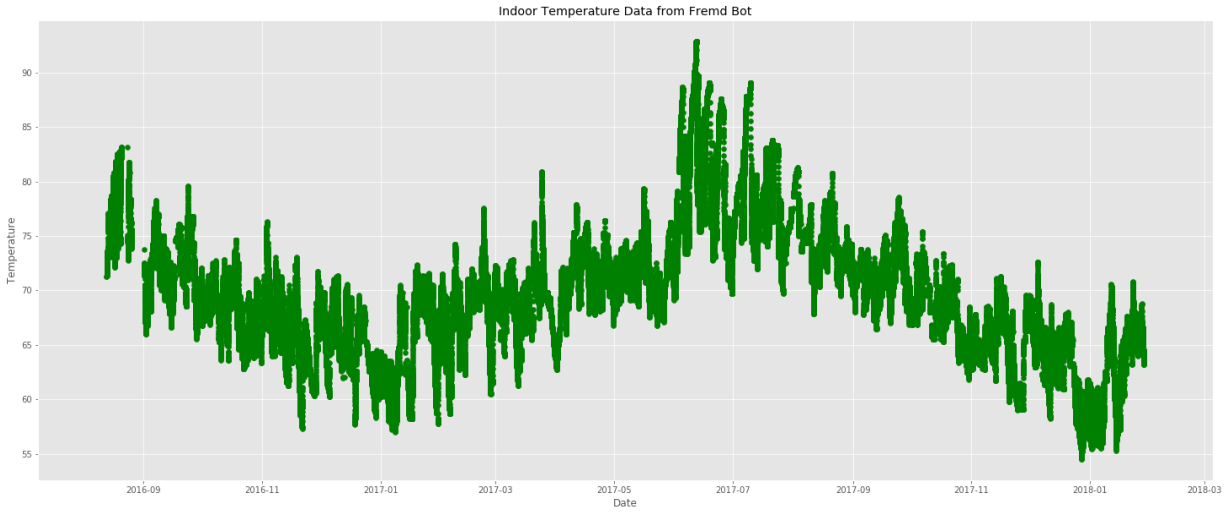
	date_time	temp
75819	2017-06-12 09:06:00 UTC	92.8616
75820	2017-06-12 09:11:00 UTC	92.8616
75821	2017-06-12 09:16:01 UTC	92.8616
75822	2017-06-12 09:21:01 UTC	92.8616
75823	2017-06-12 09:26:02 UTC	92.8616
75824	2017-06-12 09:31:03 UTC	92.8616
75825	2017-06-12 09:36:03 UTC	92.8616
75826	2017-06-12 09:41:04 UTC	92.8616
75830	2017-06-12 10:01:06 UTC	92.8616
75828	2017-06-12 09:51:05 UTC	92.8616

Plot:

In [47]:
```
# Import matplotlib
import matplotlib.pyplot as plot
%matplotlib inline

# Graph details
plot.figure(2,(25,10))
plot.style.use("ggplot")  # fivethirtyeight, bmh, grayscale, dark_background, ggp
plot.title("Indoor Temperature Data from Fremd Bot")
plot.xlabel("Date")
plot.ylabel("Temperature")

#plot.scatter(x, y, s=area, c=colors, alpha=0.5)
plot.scatter(indoor_dates,indoor_clean["temp"],color="green")
plot.show()
```



Task 1

- What do you notice about this scatterplot compared to the outdoor temperature scatterplot?

Answer

- The outside tempreture were more extreme than the inside tempreture, and there is missing data within the outside data due to inactivity

Task 2

- Show the top 10 hottest temperatures recorded in Room 223
- What was the date and time of the hottest temperature recorded in Room 223?

Answer

- The highest tempreture in room 223 is 92.8616 degrees

In [48]:
```
high_temps = above_80[-10:]

# Print the first ten values
high_temps
```

Out[48]:

	date_time	temp
75819	2017-06-12 09:06:00 UTC	92.8616
75820	2017-06-12 09:11:00 UTC	92.8616
75821	2017-06-12 09:16:01 UTC	92.8616
75822	2017-06-12 09:21:01 UTC	92.8616
75823	2017-06-12 09:26:02 UTC	92.8616
75824	2017-06-12 09:31:03 UTC	92.8616
75825	2017-06-12 09:36:03 UTC	92.8616
75826	2017-06-12 09:41:04 UTC	92.8616
75830	2017-06-12 10:01:06 UTC	92.8616
75828	2017-06-12 09:51:05 UTC	92.8616

Task 3

- Show the 10 coldest temperatures recorded in Room 223
- What was the date and time of the coldest temperature recorded in Room 223?

Answer

- The lowest tempreture in room 223 is 54.5 degrees

In [49]:
```
# Create variable, store the last 10 temps
low_temps = below_50[:10]

# Print the first ten values
low_temps
```

Out[49]:

	date_time	temp
123437	2017-12-27 15:46:28 UTC	54.5000
123436	2017-12-27 15:41:28 UTC	54.5000
123434	2017-12-27 15:31:27 UTC	54.6116
123435	2017-12-27 15:36:27 UTC	54.6116
123431	2017-12-27 15:16:25 UTC	54.6116
123432	2017-12-27 15:21:26 UTC	54.7250
123433	2017-12-27 15:26:26 UTC	54.7250
123429	2017-12-27 15:06:24 UTC	54.7250
123430	2017-12-27 15:11:25 UTC	54.7250
123428	2017-12-27 15:01:24 UTC	54.8366

Task 4

- What was the average temperature in Room 223 based on the provided dataset?

Answer

- The average tempreture in room 223 is 69.89 degrees

In [50]:
```
average = np.average(new_original_data["temp"])

average
```

Out[50]:
```
69.8866966280787
```

Challenge

- Use the outdoor temperature data to determine the date and magnitude of the largest 24-hour temperature swing
  - For example, suppose that it was 50 degrees F on 1/5/17 at 5:00pm and -10 degrees F on 1/6/17 at 3:00am
  - If this were the case, then this would be a 24-hour temperature swing of -60 degrees

In [ ]: