Looking at reactions per-turn

We are analyzing the reactions linked to the debate transcript corpus on a turn-by-turn basis.

In [1]:
```
import pandas as pd
import reactions
```

Load the table.

In [10]:
```
%time t = reactions.link_reactions_to_transcript('data/reactions_oct3_4project.csv','corpora/oct3_code
t
```

```
CPU times: user 7.91 s, sys: 0.47 s, total: 8.38 s
Wall time: 8.37 s
```

Out[10]:
```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 189015 entries, 0 to 191634
Data columns:
Frame            189015  non-null values
QuestionTopic    189015  non-null values
Reaction_what    189015  non-null values
Reaction_who     189015  non-null values
Speaker          189015  non-null values
Sync'd end       189015  non-null values
Sync'd start     189015  non-null values
Time             189015  non-null values
Tone             189015  non-null values
Topic            189015  non-null values
Transcript       189015  non-null values
UserID           189015  non-null values
start            189015  non-null values
turn             189015  non-null values
dtypes: float64(5), int64(1), object(8)
```

In [60]:
```
t[['turn','Speaker','Transcript','start','Reaction_what','Reaction_who']].head(2)
```

Out[60]:

	turn	Speaker	Transcript	start	Reaction_what	Reaction_who
0	1	0	Good evening from the Magness Arena at the Uni...	01:02:01	Agree	Moderator
56861	1	0	Good evening from the Magness Arena at the Uni...	01:02:01.401000	Disagree	Moderator

In [61]:
```
t[['turn','Speaker','Transcript','start','Reaction_what','Reaction_who']].tail(2)
```

Out[61]:

	turn	Speaker	Transcript	start	Reaction_what	Reaction_who
68397	190	0	Thank you, and good night.	02:32:59.726000	Disagree	Romney
191634	190	0	Thank you, and good night.	02:32:59.840000	Agree	Romney

In [15]:
```
print t[:1]
```

```
   Frame  QuestionTopic Reaction_what Reaction_who  Speaker Sync'd end Sync'd start  \
0      9             99         Agree    Moderator        0    1:02:06      1:02:01

                         Time  Tone  Topic  \
0  2012-10-04 01:02:00.967000     0      9
```

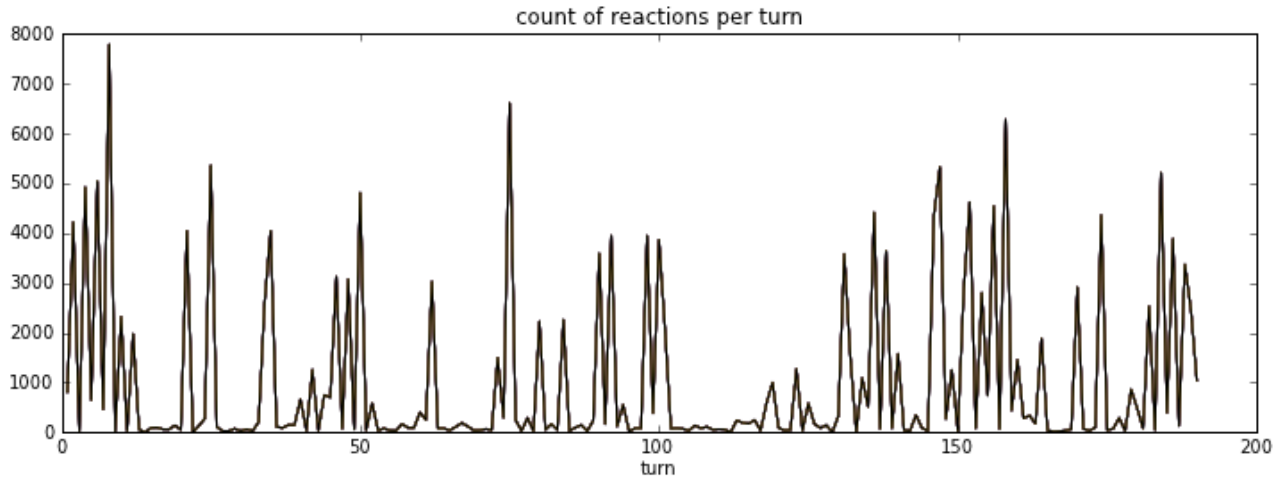
```
                                                Transcript  \
0  Good evening from the Magness Arena at the Uni...

                                                  UserID     start  turn
0  ag1zfnJlYWN0bGFicy00ciwLEgRVc2VyIiJhX2YzNTQxZW...  01:02:01     1
```

Number of reactions for each turn

In [59]:
```
t.groupby('turn').count().plot(legend=False, figsize=(12, 4), title='count of reactions per turn')
```

Out[59]: <matplotlib.axes.AxesSubplot at 0xbc85750>

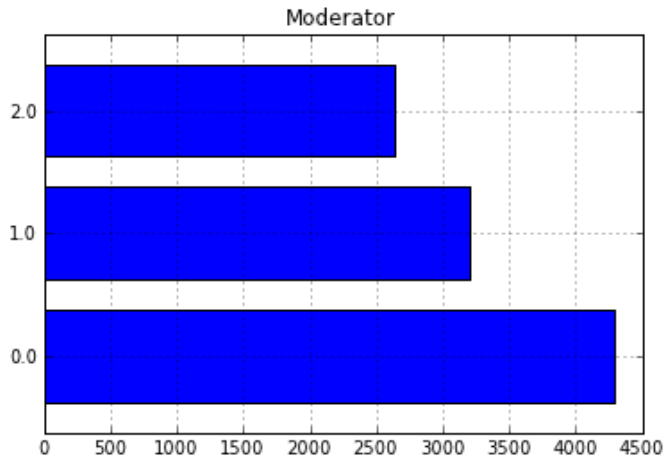


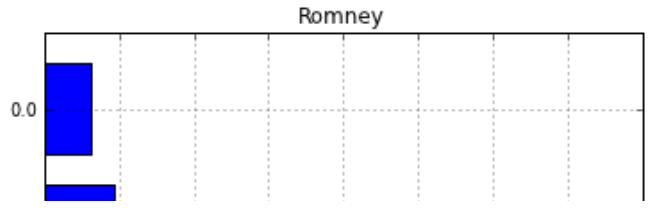
Looking at all reactions for each speaker

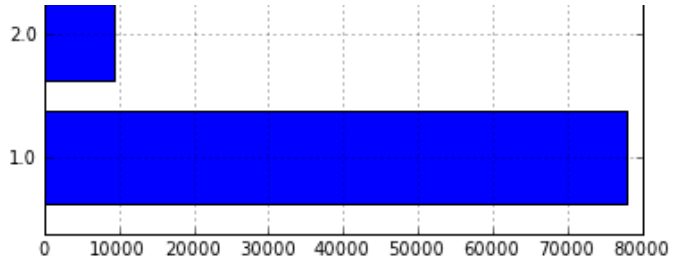
How does Speaker map to Reaction_who?

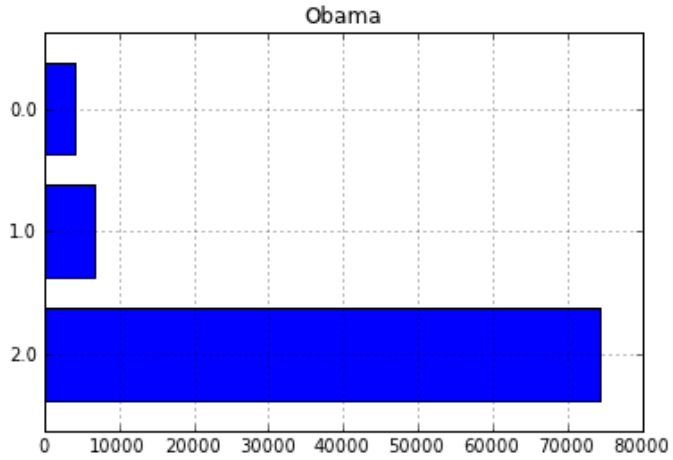
0 = Moderator **1** = Romney **2** = Obama

In [52]:
```
for s in ['Moderator','Romney','Obama']:
    t[t.Reaction_who == s].Speaker.value_counts().plot(title=s, kind='barh')
    show()
```





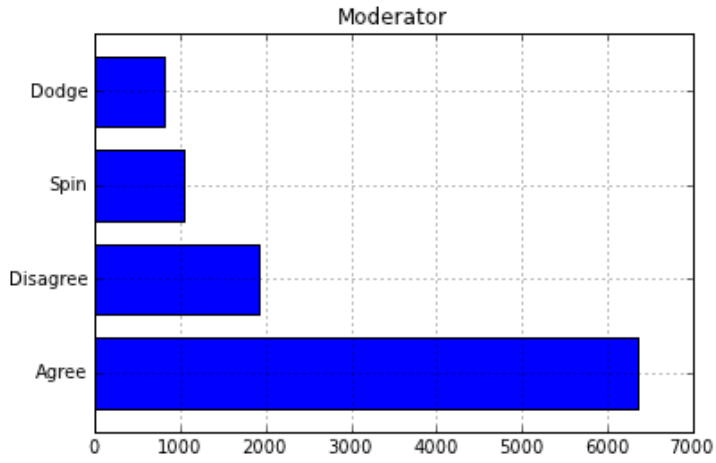


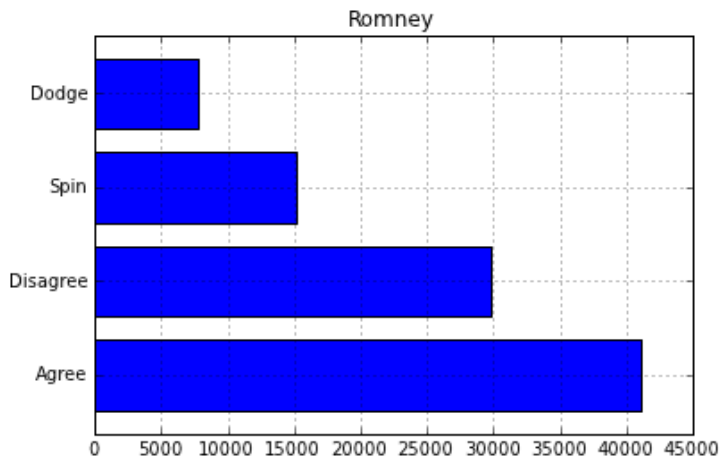


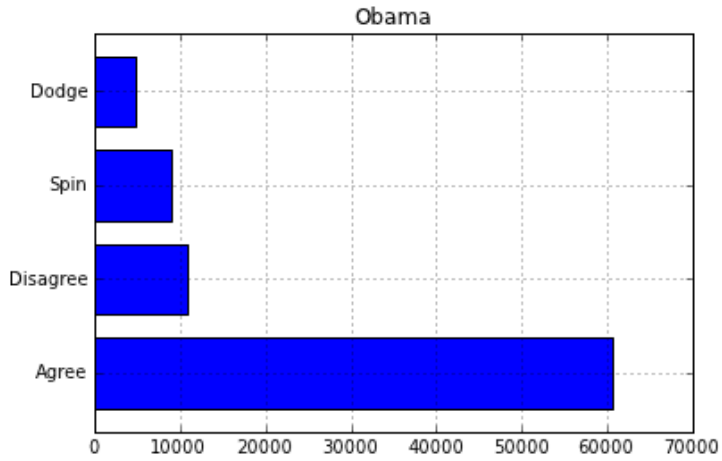
It is interesting that a lot of the time people are reacting to people who are not speaking. Why is this? This is especially true when the moderator is speaking, but perhaps that is not unexpected.

Now let's look at the the reaction data alone to see how people feel about each candidate.

In [55]:
```
for s in ['Moderator','Romney','Obama']:
    t[t.Reaction_who == s].Reaction_what.value_counts().plot(title=s, kind='barh')
    show()
```

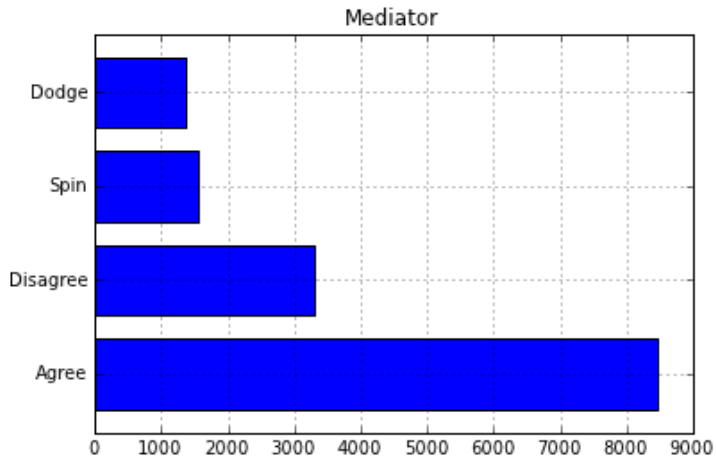


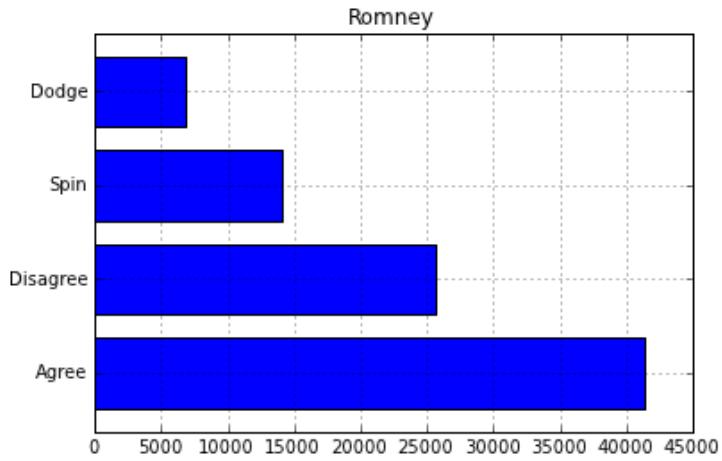


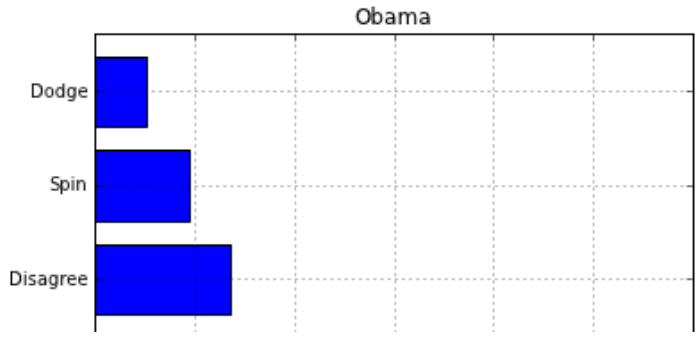


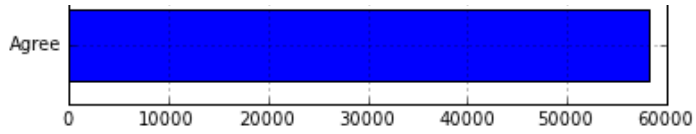
We can also look at the reactions for each candidate based on who the **transcript** says is speaking. The only difference that is obvious here is that the moderator is getting flack for some negative reactions for the candidates.

In [56]:
```
for s,n in [(0,'Mediator'),(1,'Romney'),(2,'Obama')]:
    t[t.Speaker == s].Reaction_what.value_counts().plot(title=n, kind='barh')
    show()
```









In [ ]: