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

In [3]:
```
df_project_data = pd.read_csv('C:\\temp\\workspace-CodeQL\\metadata_repositories.csv', thousands=',')

print ('Excluding projects with less than ' + str(df_project_data['num_lines'].quantile(q=0.25)) + ' lines' )
df_project_data = df_project_data[df_project_data.num_lines > df_project_data['num_lines'].quantile(q=0.25) ]
```

```
Excluding projects with less than 5016.0 lines
```

In [4]:
```
#we have a set of 72 repositories available to run queries
len(df_project_data)
```

Out[4]:
```
72
```

In [8]:
```
#merging all result files into one data frame
df_all_atoms = pd.DataFrame()
for r in results:
    df_temp = pd.read_csv(result_folder + '\\' + r)
    df_all_atoms = df_all_atoms.append(df_temp)
```

In [10]:
```
#number of repos containing atoms
df_atoms_num_repo = df_all_atoms.groupby('atom_name').full_name.nunique()
df_atoms_num_repo = df_atoms_num_repo.sort_values(0).to_frame().reset_index()
df_atoms_num_repo.columns = ['atom_name','num_repo']
df_atoms_num_repo['rate_repo'] = df_atoms_num_repo['num_repo']/len(df_project_data)
```

In [25]:
```
df_atoms_num_repo
```

Out[25]:

	atom_name	num_repo	rate_repo
0	CommaOperator	11	0.152778
1	ArithmeticasLogic	17	0.236111
2	AutomaticSemicolonInsertion	24	0.333333
3	LogicasControlFlow	41	0.569444
4	AssignmentasValue	59	0.819444
5	PreIncremenExpression	60	0.833333
6	PostIncrementExpression	65	0.902778
7	OmittedCurlyBraces	66	0.916667
8	ImplicitPredicate	72	1.000000
9	TernaryOperator	72	1.000000

In [12]:
```
df_all_atoms = df_all_atoms[df_all_atoms['full_name'].isin(df_project_data['full_name'])]
```

In [14]:
```
# number of times each atom was found on all repositories
df_atoms_count = df_all_atoms.groupby('atom_name').size().reindex()
df_atoms_count = df_atoms_count.sort_values(0).to_frame().reset_index()
df_atoms_count.columns = ['atom_name','ocurrence']
```

In [15]:
```
#number of repos each atom was found
df_atoms_num_repo = df_all_atoms.groupby('atom_name').full_name.nunique()
df_atoms_num_repo = df_atoms_num_repo.sort_values(0).to_frame().reset_index()
df_atoms_num_repo.columns = ['atom_name','num_repo']
df_atoms_num_repo['rate_repo'] = df_atoms_num_repo['num_repo']/len(df_project_data)
```

In [51]:
```
## number of all atoms found on all repositories
len(df_all_atoms)
```

Out[51]:
```
364873
```

In [17]:
```
total_lines = sum(df_project_data['num_lines'])
```

In [18]:
```
total_lines
```

Out[18]:
```
8023128.0
```

In [19]:
```
df_temp1 = df_all_atoms.groupby(['atom_name','full_name']).size().to_frame().reset_index()
df_temp1.columns = ['atom_name','full_name','ocurrence']
df_atoms_freq = pd.merge(left=df_temp1, right=df_project_data, how='left', left_on='full_name', right_on='full_name'
)
df_atoms_freq = df_atoms_freq.groupby('atom_name').agg({
        'ocurrence' : 'sum',
        'num_lines' : 'sum'
})
df_atoms_freq.reset_index(inplace=True)
df_atoms_freq['rate_lines'] = df_atoms_freq['ocurrence']/(total_lines/1000)
```

In [21]:
```
df_stats_freq = pd.merge(left=df_atoms_num_repo, right=df_atoms_freq, left_on='atom_name', right_on='atom_name')
```

In [22]:
```
df_stats_freq[['atom_name','rate_lines']]
```

Out[22]:

	atom_name	rate_lines
0	CommaOperator	0.028916
1	ArithmeticasLogic	0.020566
2	AutomaticSemicolonInsertion	0.132492
3	LogicasControlFlow	0.948508
4	AssignmentasValue	1.023790
5	PreIncremenExpression	1.039121
6	PostIncrementExpression	5.624490
7	OmittedCurlyBraces	6.606899
8	ImplicitPredicate	19.890123
9	TernaryOperator	10.162745

In [23]:
```
df_stats_freq_chart = df_stats_freq[['atom_name','rate_lines']]

df_stats_freq_chart.sort_values(ascending=False, inplace=True,by='rate_lines')
```

```
[redacted]s\python\python37_64\lib\site-packages\ipykernel_launcher.py:3: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: http://pandas.pydata.org/pandas-docs/stable/indexing.html#indexing-view-versus-copy
  This is separate from the ipykernel package so we can avoid doing imports until
```

In [24]:
```
#sns.set(style="whitegrid")

f, ax = plt.subplots()
sns.set_color_codes("pastel")
sns.set(style="whitegrid")
sns.despine(top=True, right=True, left=True, bottom=True)

sns.barplot(x="rate_lines", y="atom_name", data=df_stats_freq_chart, orient='horizontal',color="b",)
ax.set_xlabel('Occurrence/KLOC')
ax.set(xticks=[])
ax.set_ylabel('Atom')

for p in ax.patches:
    width = p.get_width()
    ax.text(12 ,
            p.get_y() + 0.7 ,
            '{:1.2f}'.format(width),
            ha="center")
plt.tight_layout()


f.savefig('chart_ocurrence_kloc.png', format='png',dpi=200)
f.savefig('chart_ocurrence_kloc.svg', format='svg',dpi=200)
```