In [1]:
```
# This Python 3 environment comes with many helpful analytics libraries installed
# It is defined by the kaggle/python Docker image: https://github.com/kaggle/docker-python
# For example, here's several helpful packages to load

# You can write up to 20GB to the current directory (/kaggle/working/) that gets preserved as output when you create a version using "Save & Run All"
# You can also write temporary files to /kaggle/temp/, but they won't be saved outside of the current session
```

1. Data Details

Columns

`repositories` - the name of the repository (Format - github_username/repository_name)

`stars_count` - stars count of the repository

`forks_count` - fork count of the repository

`issues_count` - active/opened issues in the repository

`pull_requests` - pull requests opened in the repository

`contributors` - contributors contribute to the project so far

`language` - primary language used in the project

In [2]:
```
import numpy as np # linear algebra
import pandas as pd # data processing, CSV file I/O (e.g. pd.read_csv)

# Load data set.
github_data = pd.read_csv('/kaggle/input/github-dataset/github_dataset.csv')

# Check the columns.
github_data.columns
```

Out[2]:
```
Index(['repositories', 'stars_count', 'forks_count', 'issues_count',
       'pull_requests', 'contributors', 'language'],
      dtype='object')
```

In [3]:
```
# Show first five rows as an example.
github_data.head()
```

Out[3]:

	repositories	stars_count	forks_count	issues_count	pull_requests	contributors	language
0	octocat/Hello-World	0	0	612	316	2	NaN
1	EddieHubCommunity/support	271	150	536	6	71	NaN
2	ethereum/aleth	0	0	313	27	154	C++
3	localstack/localstack	0	0	290	30	434	Python
4	education/classroom	0	589	202	22	67	Ruby

Nominal Attributes

`repositories` - the name of the repository (Format - github_username/repository_name)

`language` - primary language used in the project

Frequency -- repositories

In [4]:
```
github_data['repositories'].value_counts()
```

Out[4]:
```
kameshsampath/ansible-role-rosa-demos          2
aloisdeniel/bluff                              2
antoniaandreou/github-slideshow                2
jgthms/bulma-start                             2
artkirienko/hlds-docker-dproto                 2
                                              ..
WhiteHouse/CIOmanagement                       1
0xCaso/defillama-telegram-bot                  1
ethereum/blake2b-py                            1
openfoodfacts/folksonomy_mobile_experiment     1
gamemann/All_PropHealth                        1
Name: repositories, Length: 972, dtype: int64
```

Frequency -- language

In [5]:
```
github_data['language'].value_counts()
```

Out[5]:
```
JavaScript          253
Python              155
HTML                 72
Java                 44
CSS                  37
TypeScript           37
Dart                 36
C++                  29
```

```
Jupyter Notebook     29
Ruby                 28
C                    26
Shell                25
PHP                  16
Go                   15
Rust                 10
Swift                10
C#                    8
Objective-C           8
Kotlin                7
Makefile              6
Jinja                 5
SCSS                  4
CoffeeScript          3
Perl                  3
Dockerfile            3
Solidity              3
AutoHotkey            3
Hack                  2
Pawn                  2
CodeQL                2
PowerShell            2
Assembly              2
Vim Script            2
Vue                   2
Elixir                2
Gherkin               1
QMake                 1
CMake                 1
Oz                    1
Cuda                  1
QML                   1
ActionScript          1
Roff                  1
HCL                   1
R                     1
PureBasic             1
Smarty                1
Less                  1
Svelte                1
Haskell               1
SourcePawn            1
Name: language, dtype: int64
```

Numeric Attributes

`stars_count` - stars count of the repository

`forks_count` - fork count of the repository

`issues_count` - active/opened issues in the repository

`pull_requests` - pull requests opened in the repository

`contributors` - contributors contribute to the project so far

In [6]:
```
numeric_github_data = pd.DataFrame(github_data, columns=['stars_count', 'forks_count', 'issues_count', 'pull_requests', 'contributors'])
numeric_github_data.head()
```

Out[6]:

	stars_count	forks_count	issues_count	pull_requests	contributors
0	0	0	612	316	2
1	271	150	536	6	71
2	0	0	313	27	154
3	0	0	290	30	434
4	0	589	202	22	67

In [7]:
```
github_data['stars_count'].describe()
```

Out[7]:
```
count    1052.000000
mean       81.976236
std       170.403116
min         0.000000
25%         1.000000
50%        12.000000
75%        65.250000
max       995.000000
Name: stars_count, dtype: float64
```

In [8]:
```
numeric_describe = numeric_github_data.describe()
numeric_describe
```

Out[8]:

	stars_count	forks_count	issues_count	pull_requests	contributors
count	1052.000000	1052.000000	1052.000000	1052.000000	1052.000000
mean	81.976236	53.884981	8.656844	4.374525	8.364068
std	170.403116	127.699729	32.445154	27.913732	37.511807
min	0.000000	0.000000	1.000000	0.000000	0.000000
25%	1.000000	1.000000	1.000000	0.000000	0.000000
50%	12.000000	6.000000	2.000000	0.000000	2.000000
75%	65.250000	38.250000	6.000000	2.000000	4.000000
max	995.000000	973.000000	612.000000	567.000000	658.000000

Five Number Summary -- `stars_count`, `forks_count`, `issues_count`, `pull_requests`, `contributors`

In [9]:
```
numeric_describe.loc[['mean', '25%', '50%', '75%', 'max']]
```

Out[9]:

	stars_count	forks_count	issues_count	pull_requests	contributors
mean	81.976236	53.884981	8.656844	4.374525	8.364068
25%	1.000000	1.000000	1.000000	0.000000	0.000000
50%	12.000000	6.000000	2.000000	0.000000	2.000000
75%	65.250000	38.250000	6.000000	2.000000	4.000000
max	995.000000	973.000000	612.000000	567.000000	658.000000

Missing Value Count

In [10]:
```
NaN_counts = github_data.isna().sum()
NaN_counts = pd.DataFrame(NaN_counts, columns=['NaN_counts']).T
NaN_counts
```

Out[10]:

	repositories	stars_count	forks_count	issues_count	pull_requests	contributors	language
NaN_counts	0	0	0	0	0	0	145

2. Data Visualization

Histogram for Nominal Attributes

repositories

In [11]:
```
import seaborn as sns
import matplotlib.pyplot as plt
```

In [12]:
```
repo_counts = pd.DataFrame(github_data['repositories'].value_counts())
repo_counts
```

Out[12]:

	repositories
kameshsampath/ansible-role-rosa-demos	2
aloisdeniel/bluff	2
antoniaandreou/github-slideshow	2
jgthms/bulma-start	2
artkirienko/hlds-docker-dproto	2
...	...
WhiteHouse/CIOmanagement	1
0xCaso/defillama-telegram-bot	1
ethereum/blake2b-py	1
openfoodfacts/folksonomy_mobile_experiment	1
gamemann/All_PropHealth	1

972 rows × 1 columns

In [13]:
```
repo_counts.index
```

Out[13]:
```
Index(['kameshsampath/ansible-role-rosa-demos', 'aloisdeniel/bluff',
       'antoniaandreou/github-slideshow', 'jgthms/bulma-start',
       'artkirienko/hlds-docker-dproto', 'artkirienko/int-null-even',
       'KrauseFx/dotfiles', 'carloscuesta/gitmoji', 'divyamagwl/Depocalypse',
       'ritwickdey/Cake-Shop',
       ...
       'trailofbits/circuitous-benchmarks', 'Clueless-Community/Datasets',
       'ethereum/beacon_chain', 'openfoodfacts/eu-food-data',
       'jonfroehlich/jonfroehlich.github.io', 'WhiteHouse/CIOmanagement',
       '0xCaso/defillama-telegram-bot', 'ethereum/blake2b-py',
       'openfoodfacts/folksonomy_mobile_experiment',
       'gamemann/All_PropHealth'],
      dtype='object', length=972)
```

In [14]:
```
repo_counts.T
```

Out[14]:

	kameshsampath/ansible-role-rosa-demos	aloisdeniel/bluff	antoniaandreou/github-slideshow	jgthms/bulma-start	artkirienko/hlds-docker-dproto	artkirienko/int-null-even	KrauseFx/dotfiles	carloscuesta/gitmoji	divyamagwl/Depocalypse	ritwickdey/Cake-Shop	..
repositories	2	2	2	2	2	2	2	2	2	2	..

1 rows × 972 columns

In [15]:
```
language_counts = pd.DataFrame(github_data['language'].value_counts()).sort_values(by='language', ascending=True).rename(columns={'language': 'language_cour
language_counts
```

Out[15]:

	language_count
SourcePawn	1
Gherkin	1
QMake	1
Oz	1
Cuda	1
QML	1
ActionScript	1
CMake	1
HCL	1
Roff	1
Svelte	1
Less	1
Haskell	1
Smarty	1
PureBasic	1
R	1
Elixir	2
Vim Script	2
Assembly	2
PowerShell	2
CodeQL	2
Pawn	2
Hack	2
Vue	2
AutoHotkey	3

Solidity	3
Perl	3
CoffeeScript	3
Dockerfile	3
SCSS	4
Jinja	5
Makefile	6
Kotlin	7
Objective-C	8
C#	8
Swift	10
Rust	10
Go	15
PHP	16
Shell	25
C	26
Ruby	28
Jupyter Notebook	29
C++	29
Dart	36
TypeScript	37
CSS	37
Java	44
HTML	72
Python	155
JavaScript	253

In [16]:
```
language_counts.index
```

Out[16]:
```
Index(['SourcePawn', 'Gherkin', 'QMake', 'Oz', 'Cuda', 'QML', 'ActionScript',
       'CMake', 'HCL', 'Roff', 'Svelte', 'Less', 'Haskell', 'Smarty',
       'PureBasic', 'R', 'Elixir', 'Vim Script', 'Assembly', 'PowerShell',
       'CodeQL', 'Pawn', 'Hack', 'Vue', 'AutoHotkey', 'Solidity', 'Perl',
       'CoffeeScript', 'Dockerfile', 'SCSS', 'Jinja', 'Makefile', 'Kotlin',
       'Objective-C', 'C#', 'Swift', 'Rust', 'Go', 'PHP', 'Shell', 'C', 'Ruby',
       'Jupyter Notebook', 'C++', 'Dart', 'TypeScript', 'CSS', 'Java', 'HTML',
       'Python', 'JavaScript'],
      dtype='object')
```

In [17]:
```
pd.DataFrame(github_data['language'].value_counts()).T
```

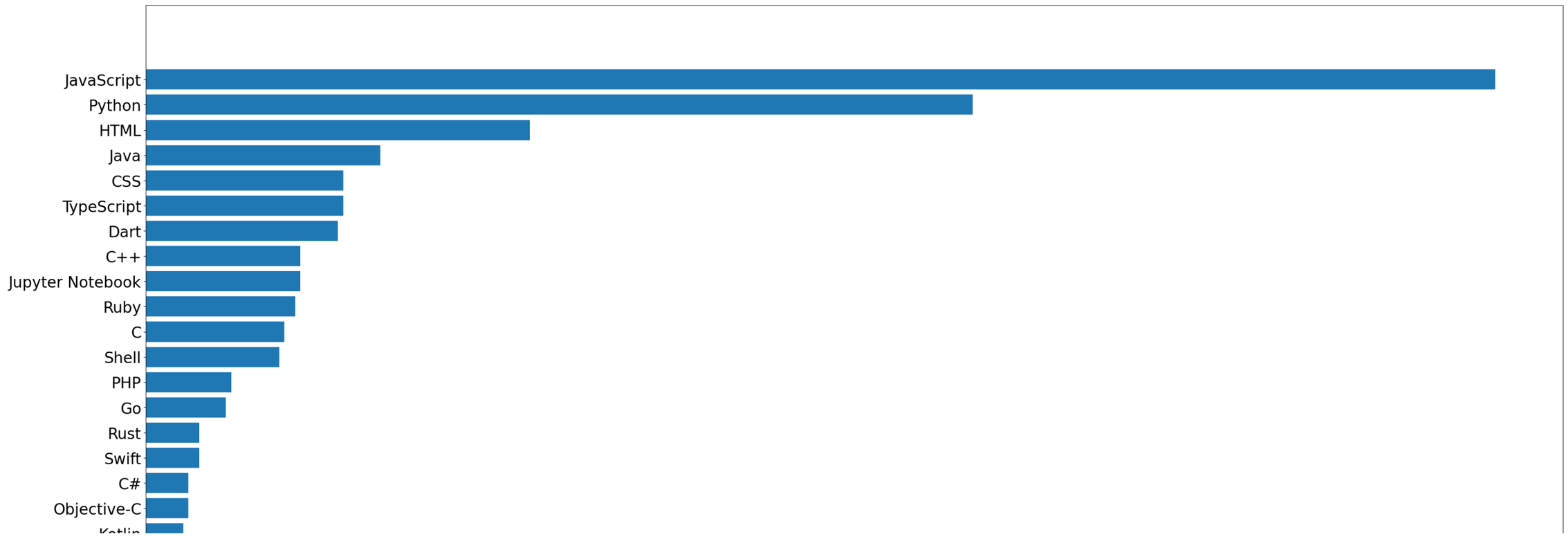
Out[17]:

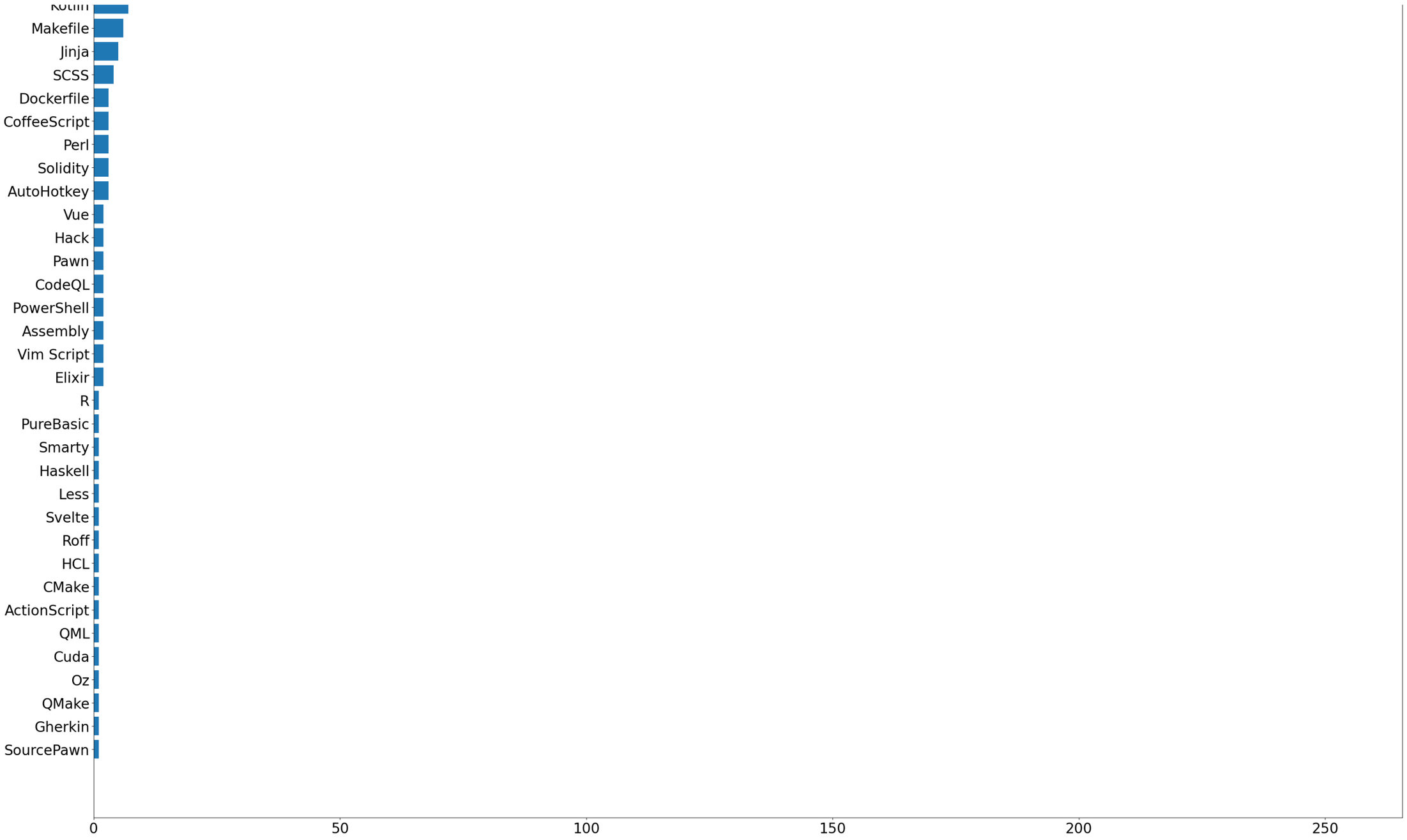
	JavaScript	Python	HTML	Java	CSS	TypeScript	Dart	C++	Jupyter Notebook	Ruby	...	ActionScript	Roff	HCL	R	PureBasic	Smarty	Less	Svelte	Haskell	SourcePawn
language	253	155	72	44	37	37	36	29	29	28	...	1	1	1	1	1	1	1	1	1	1

1 rows × 51 columns

In [18]:
```
plt.figure(figsize=(40, 40))
plt.yticks(fontsize=24)
plt.xticks(fontsize=24)
plt.barh(language_counts.index, width=language_counts['language_count'])
```

Out[18]:
```
<BarContainer object of 51 artists>
```




Kotlin
Makefile
Jinja
SCSS
Dockerfile
CoffeeScript
Perl
Solidity
AutoHotkey
Vue
Hack
Pawn
CodeQL
PowerShell
Assembly
Vim Script
Elixir
R
PureBasic
Smarty
Haskell
Less
Svelte
Roff
HCL
CMake
ActionScript
QML
Cuda
Oz
QMake
Gherkin
SourcePawn
0
50
100
150
200
250

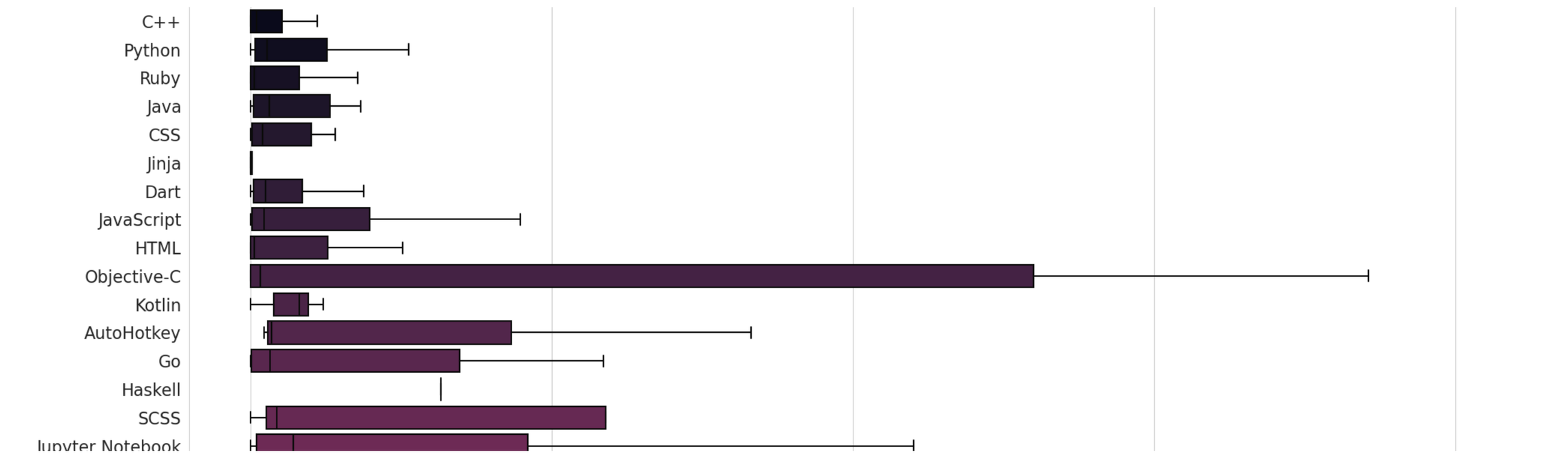
In [19]:
```
language_list = [np.nan]
encode_language_list = []
nan = np.nan
for language in github_data['language']:
    if language not in language_list:
        language_list.append(language)
    language = language_list.index(language)
encode_language_list.append(language)
print(language_list)
print(len(language_list))
```

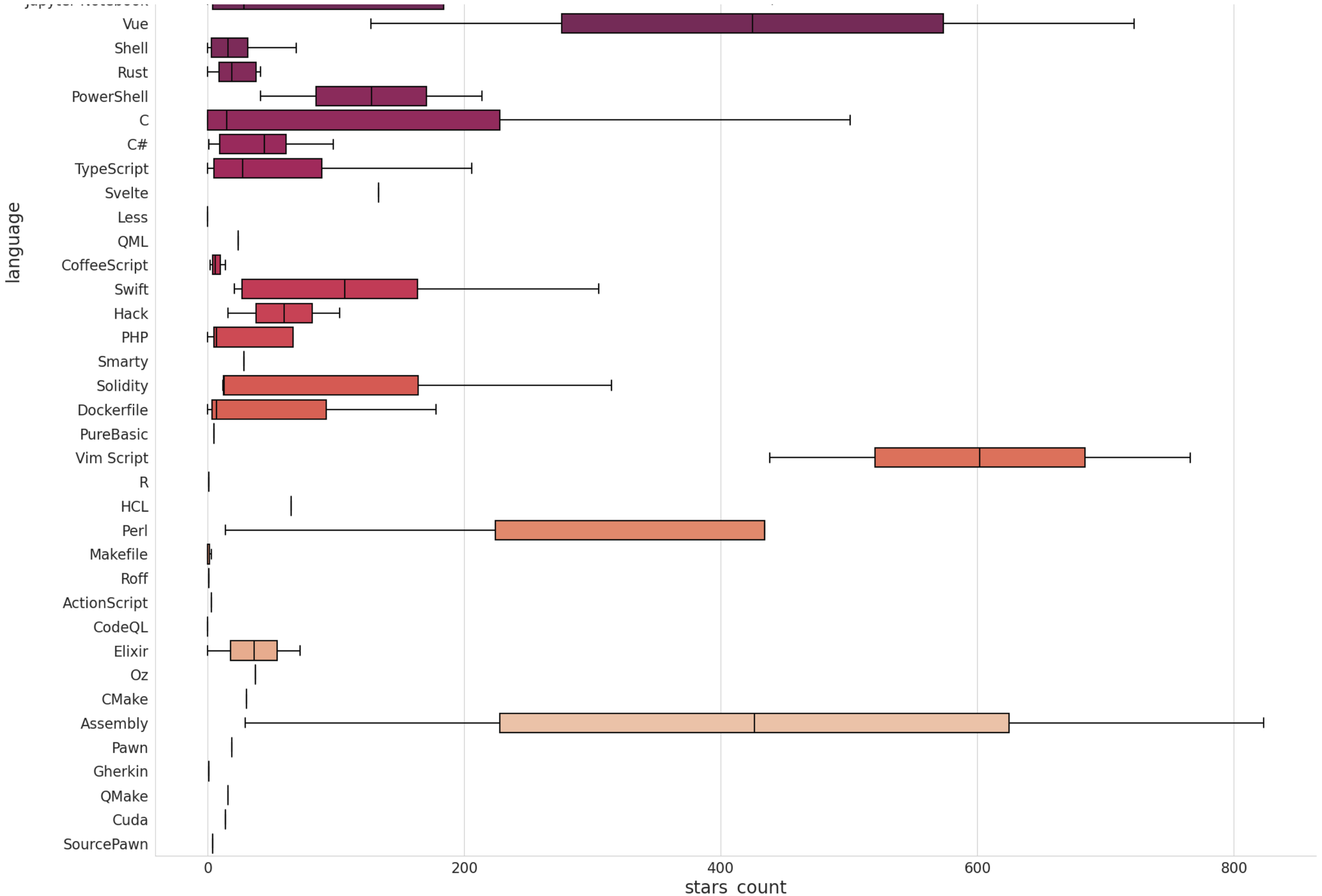
```
[nan, 'C++', 'Python', 'Ruby', 'Java', 'CSS', 'Jinja', 'Dart', 'JavaScript', 'HTML', 'Objective-C', 'Kotlin', 'AutoHotkey', 'Go', 'Haskell', 'SCSS', 'Jupy
ter Notebook', 'Vue', 'Shell', 'Rust', 'PowerShell', 'C', 'C#', 'TypeScript', 'Svelte', 'Less', 'QML', 'CoffeeScript', 'Swift', 'Hack', 'PHP', 'Smarty', '
Solidity', 'Dockerfile', 'PureBasic', 'Vim Script', 'R', 'HCL', 'Perl', 'Makefile', 'Roff', 'ActionScript', 'CodeQL', 'Elixir', 'Oz', 'CMake', 'Assembly',
'Pawn', 'Gherkin', 'QMake', 'Cuda', 'SourcePawn']
52
```

In [20]:
```
# github_data_encoded = github_data.insert(loc=len(github_data.columns), column='language_encoded', value=[l for l in encode_language_list])
```

In [21]:
```
import seaborn as sns
import matplotlib.pyplot as plt
```

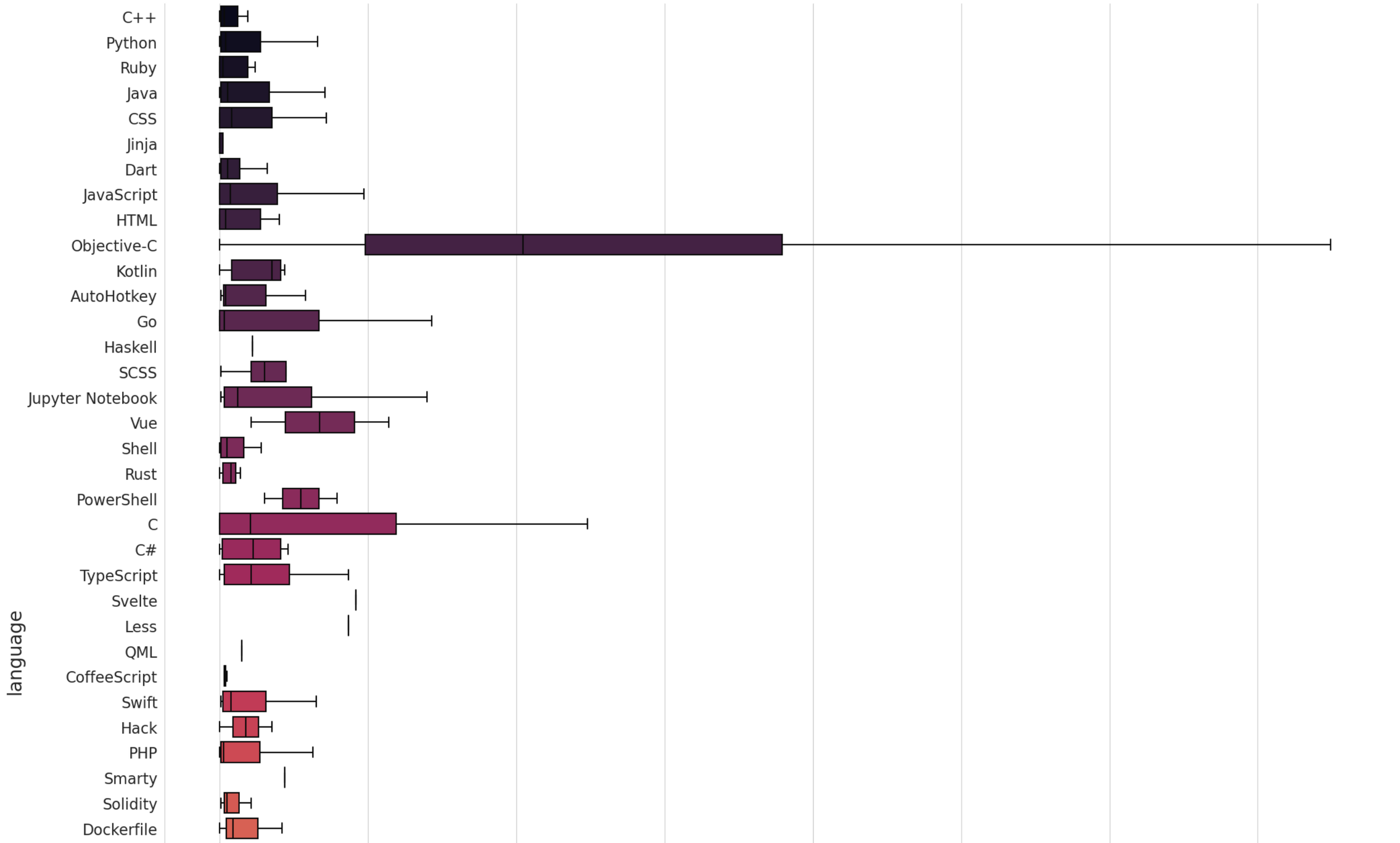
In [22]:
```
sns.set_style("whitegrid")
stars_count_box = sns.catplot(data=github_data, kind='box', y='language', x='stars_count', height=20, palette='rocket', sym='')
plt.yticks(fontsize=16)
plt.xticks(fontsize=16)
plt.ylabel('language', fontsize=20, )
plt.xlabel('stars_count', fontsize=20)
plt.show()
```

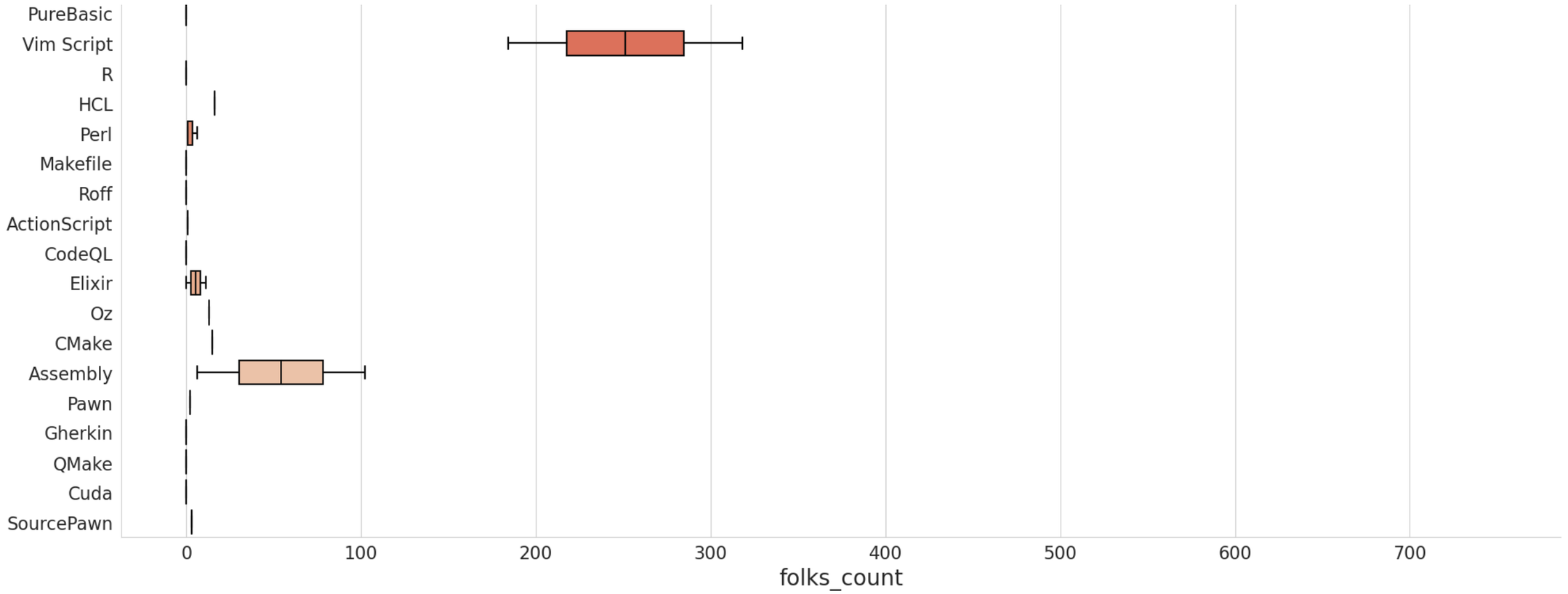




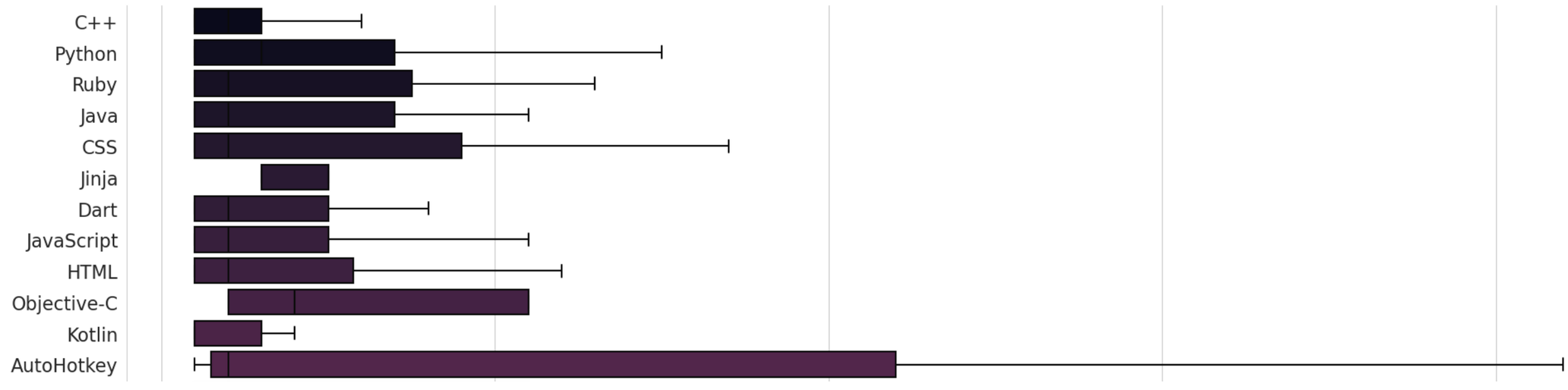
In [23]: sns.set_style("whitegrid")

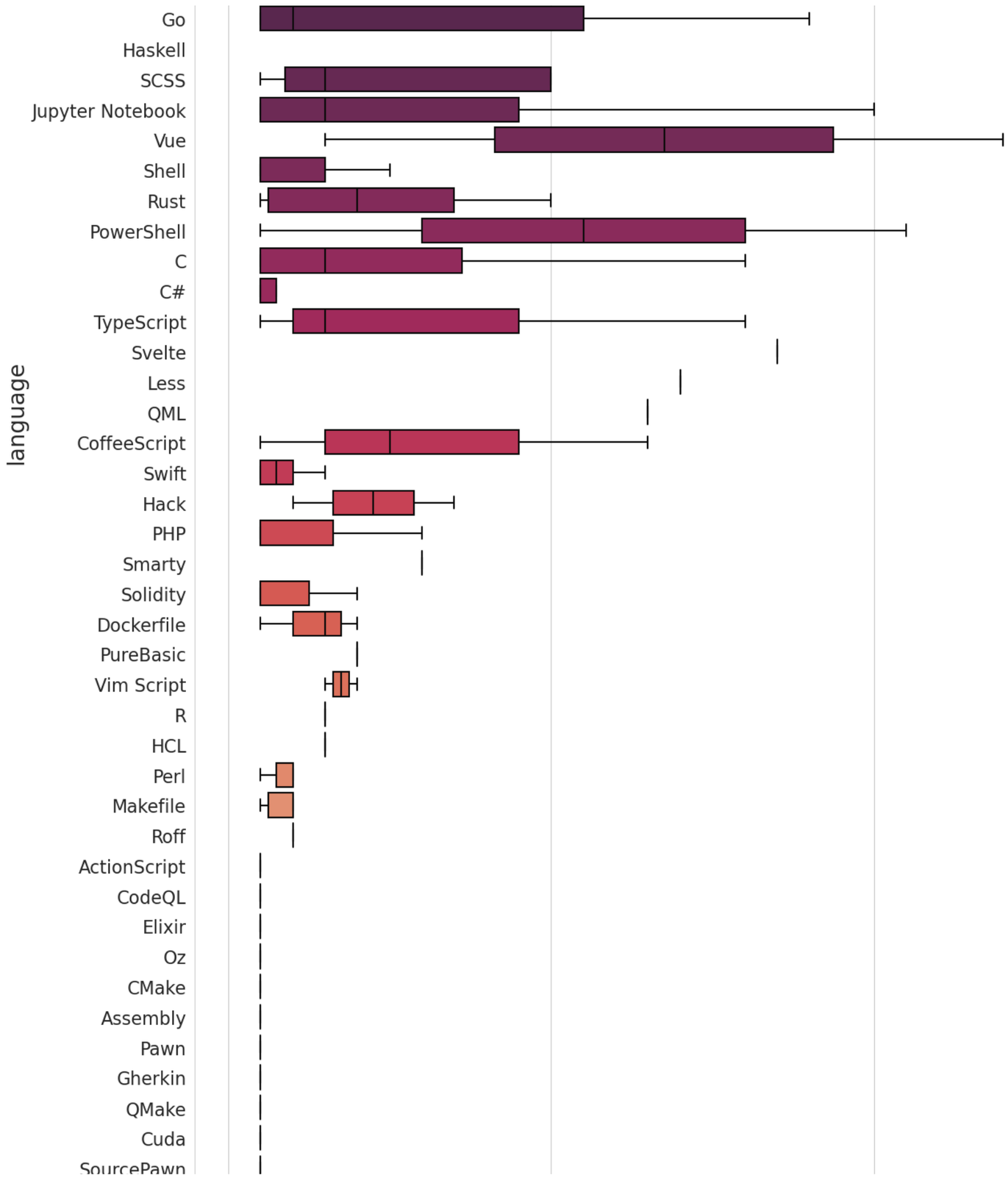
In [23]:
```
sns.set_style( 'whitegrid' )
forks_count_box = sns.catplot(data=github_data, kind='box', y='language', x='forks_count', height=20, palette='rocket', sym='')
plt.yticks(fontsize=16)
plt.xticks(fontsize=16)
plt.ylabel('language', fontsize=20, )
plt.xlabel('folks_count', fontsize=20)
plt.show()
```



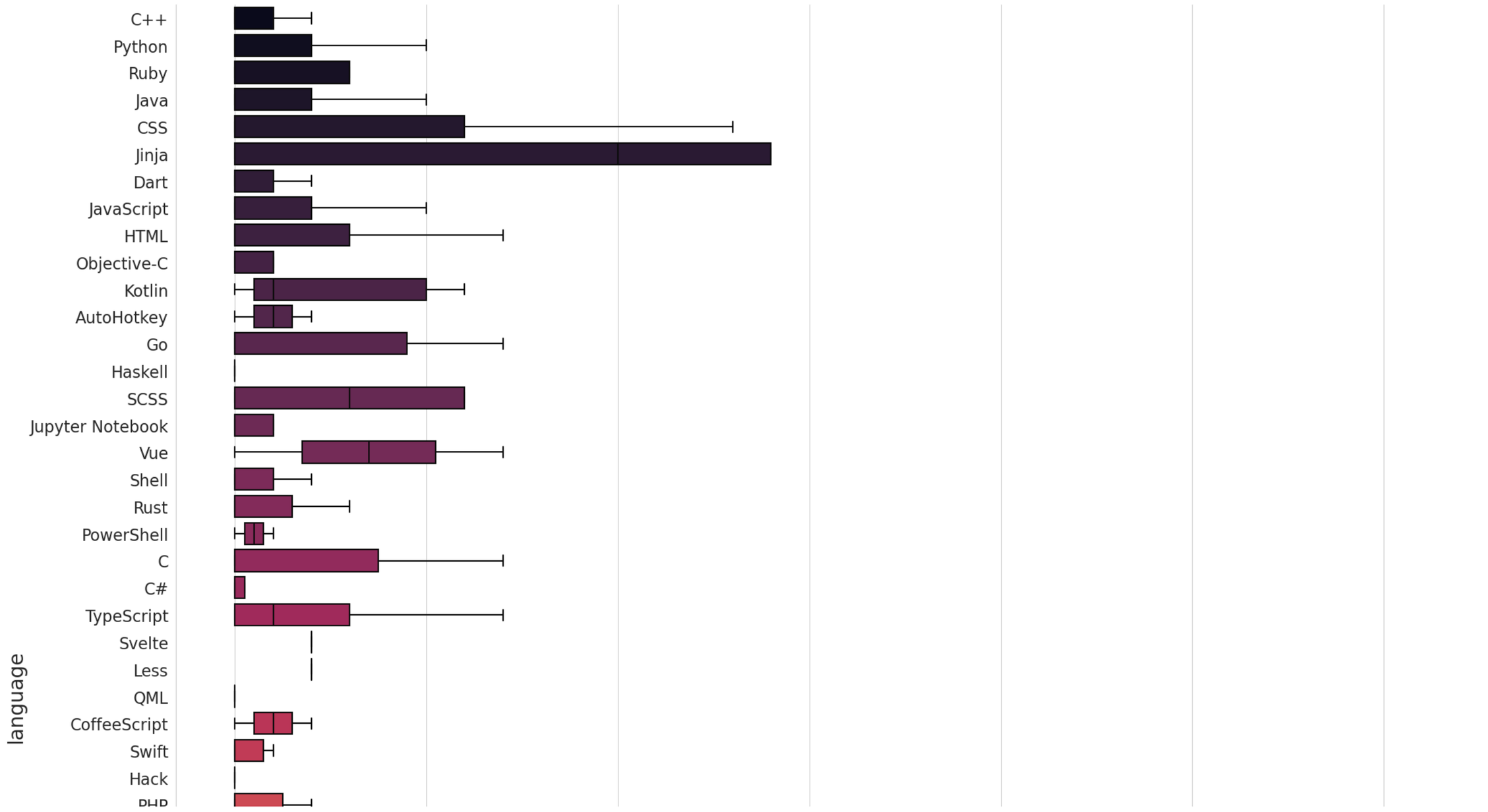


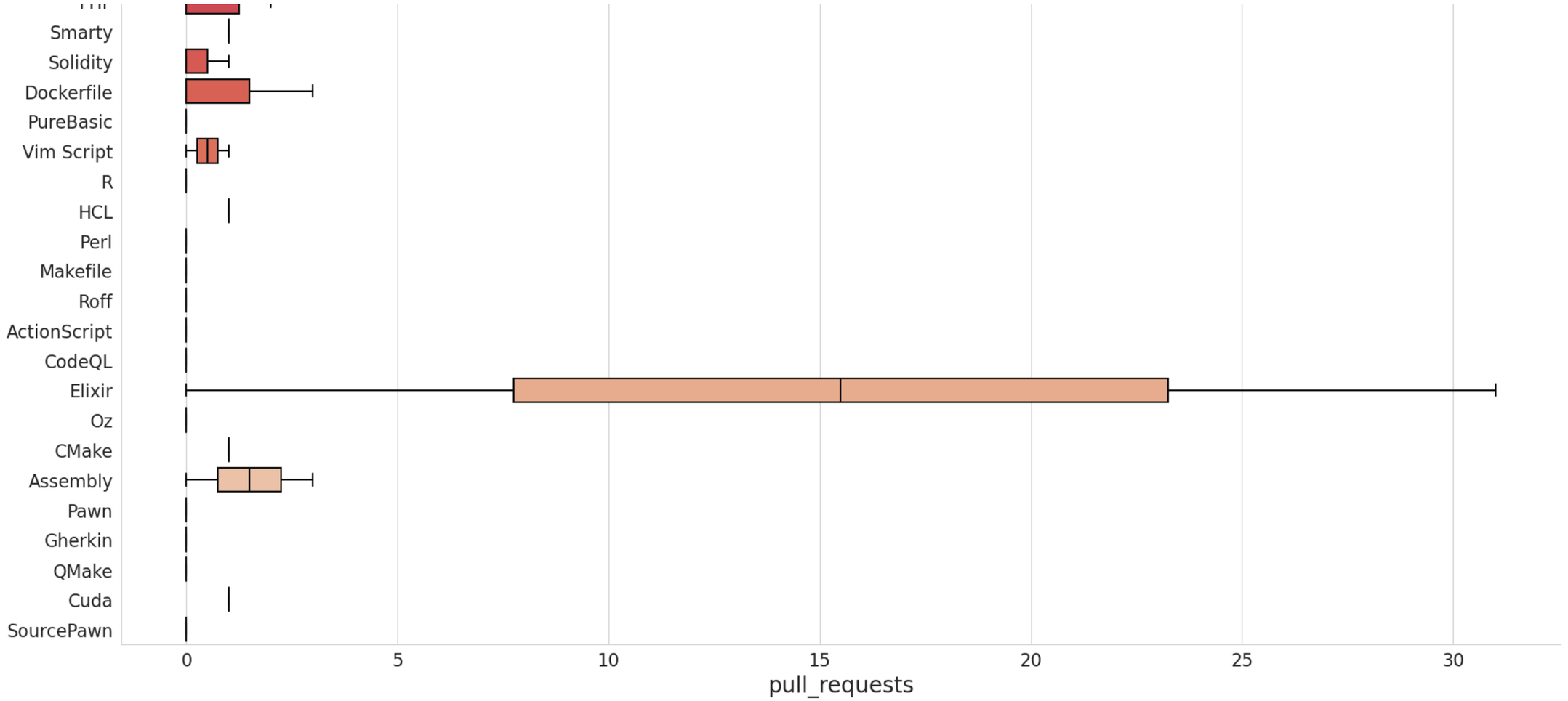
In [24]:
```
sns.set_style("whitegrid")
issues_count_box = sns.catplot(data=github_data, kind='box', y='language', x='issues_count', height=20, palette='rocket', sym='')
plt.yticks(fontsize=16)
plt.xticks(fontsize=16)
plt.ylabel('language', fontsize=20, )
plt.xlabel('issues_count', fontsize=20)
plt.show()
```



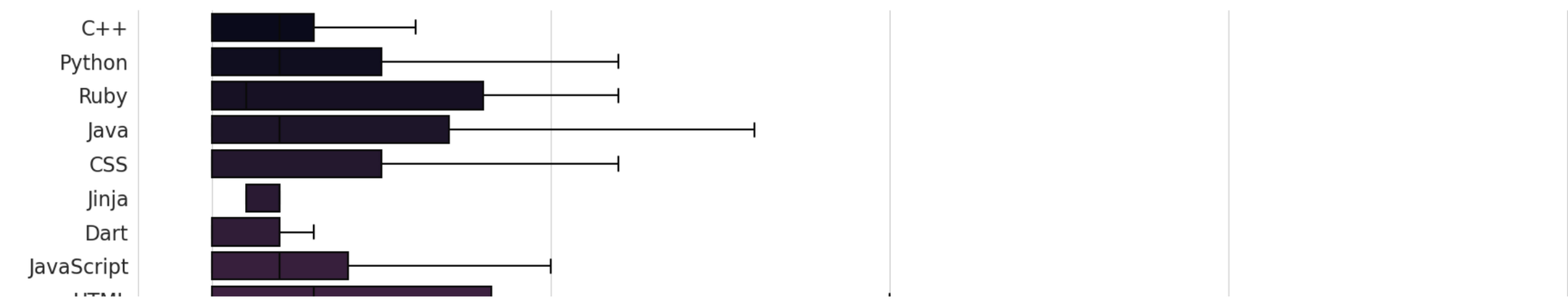

language
Go
Haskell
SCSS
Jupyter Notebook
Vue
Shell
Rust
PowerShell
C
C#
TypeScript
Svelte
Less
QML
CoffeeScript
Swift
Hack
PHP
Smarty
Solidity
Dockerfile
PureBasic
Vim Script
R
HCL
Perl
Makefile
Roff
ActionScript
CodeQL
Elixir
Oz
CMake
Assembly
Pawn
Gherkin
QMake
Cuda
SourcePawn

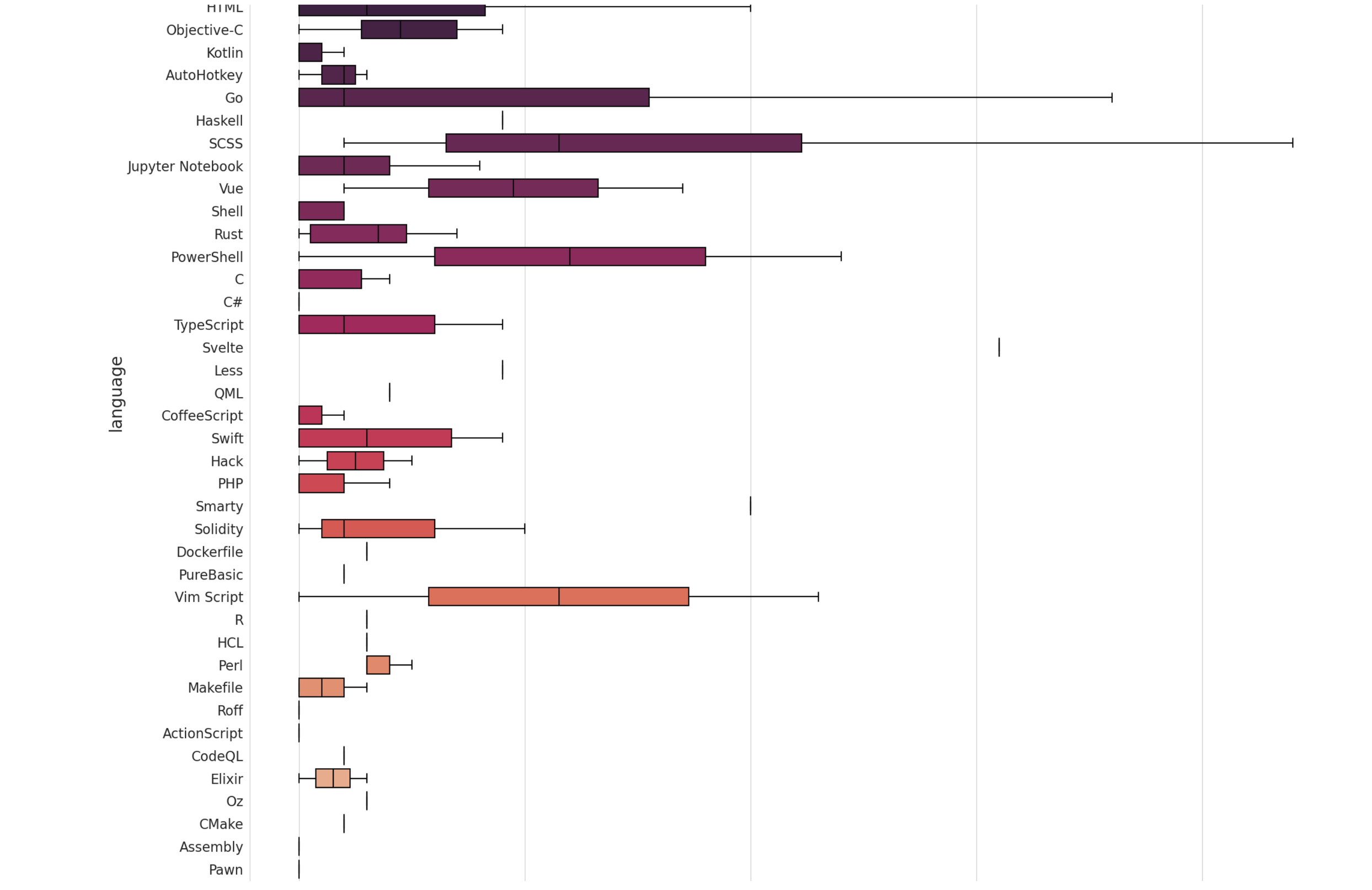
In [25]:
```
sns.set_style("whitegrid")
pull_requests_box = sns.catplot(data=github_data, kind='box', y='language', x='pull_requests', height=20, palette='rocket', sym='')
plt.yticks(fontsize=16)
plt.xticks(fontsize=16)
plt.ylabel('language', fontsize=20, )
plt.xlabel('pull_requests', fontsize=20)
plt.show()
```

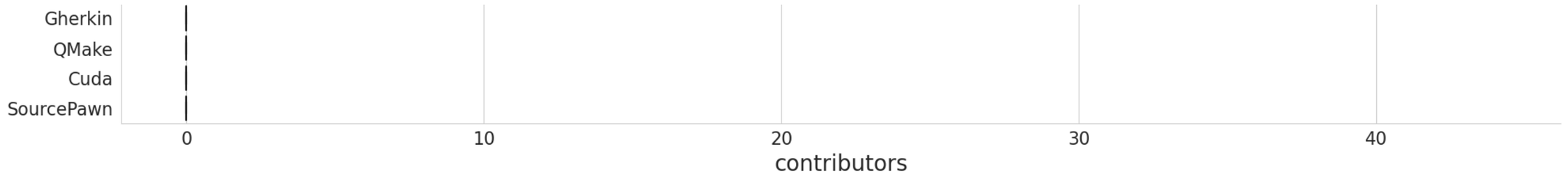




In [26]:
```
sns.set_style("whitegrid")
contributors_box = sns.catplot(data=github_data, kind='box', y='language', x='contributors', height=20, palette='rocket', sym='')
plt.yticks(fontsize=16)
plt.xticks(fontsize=16)
plt.ylabel('language', fontsize=20, )
plt.xlabel('contributors', fontsize=20)
plt.show()
```




HTML
Objective-C
Kotlin
AutoHotkey
Go
Haskell
SCSS
Jupyter Notebook
Vue
Shell
Rust
PowerShell
C
C#
TypeScript
Svelte
Less
QML
CoffeeScript
Swift
Hack
PHP
Smarty
Solidity
Dockerfile
PureBasic
Vim Script
R
HCL
Perl
Makefile
Roff
ActionScript
CodeQL
Elixir
Oz
CMake
Assembly
Pawn
language



In [27]:
```
operation_counts = pd.DataFrame(list(github_data['stars_count']), columns=['counts'])
operation_counts = pd.DataFrame(operation_counts, columns=['operation', 'counts'])
operation_counts['operation'] = 'stars_count'
operation_counts
```

Out[27]:

	operation	counts
0	stars_count	0
1	stars_count	271
2	stars_count	0
3	stars_count	0
4	stars_count	0
...	...	...
1047	stars_count	2
1048	stars_count	0
1049	stars_count	0
1050	stars_count	11
1051	stars_count	4

1052 rows × 2 columns

In [28]:
```
operation_counts = operation_counts.append(pd.DataFrame([['forks_count', c] for c in list(github_data['forks_count'])], columns=['operation', 'counts']), ig
operation_counts
```

Out[28]:

	operation	counts
0	stars_count	0
1	stars_count	271
2	stars_count	0
3	stars_count	0
4	stars_count	0
...	...	...
2099	forks_count	1
2100	forks_count	0
2101	forks_count	5
2102	forks_count	5
2103	forks_count	3

2104 rows × 2 columns

In [29]:
```
operation_counts = operation_counts.append(pd.DataFrame([['issues_count', c] for c in list(github_data['issues_count'])], columns=['operation', 'counts']),
operation_counts
```

Out[29]:

	operation	counts
0	stars_count	0
1	stars_count	271
2	stars_count	0
3	stars_count	0
4	stars_count	0
...	...	...
3151	issues_count	1
3152	issues_count	1
3153	issues_count	1
3154	issues_count	1
3155	issues_count	1

3156 rows × 2 columns

In [30]:
```
operation_counts = operation_counts.append(pd.DataFrame([['pull_requests', c] for c in list(github_data['pull_requests'])], columns=['operation', 'counts'])
operation_counts
```

Out[30]:

	operation	counts
0	stars_count	0
1	stars_count	271
2	stars_count	0
3	stars_count	0
4	stars_count	0
...	...	...
4203	pull_requests	0
4204	pull_requests	0
4205	pull_requests	1
4206	pull_requests	0
4207	pull_requests	0

4208 rows × 2 columns

In [31]:
```
operation_counts = operation_counts.append(pd.DataFrame([['contributors', c] for c in list(github_data['contributors'])], columns=['operation', 'counts']),
operation_counts
```

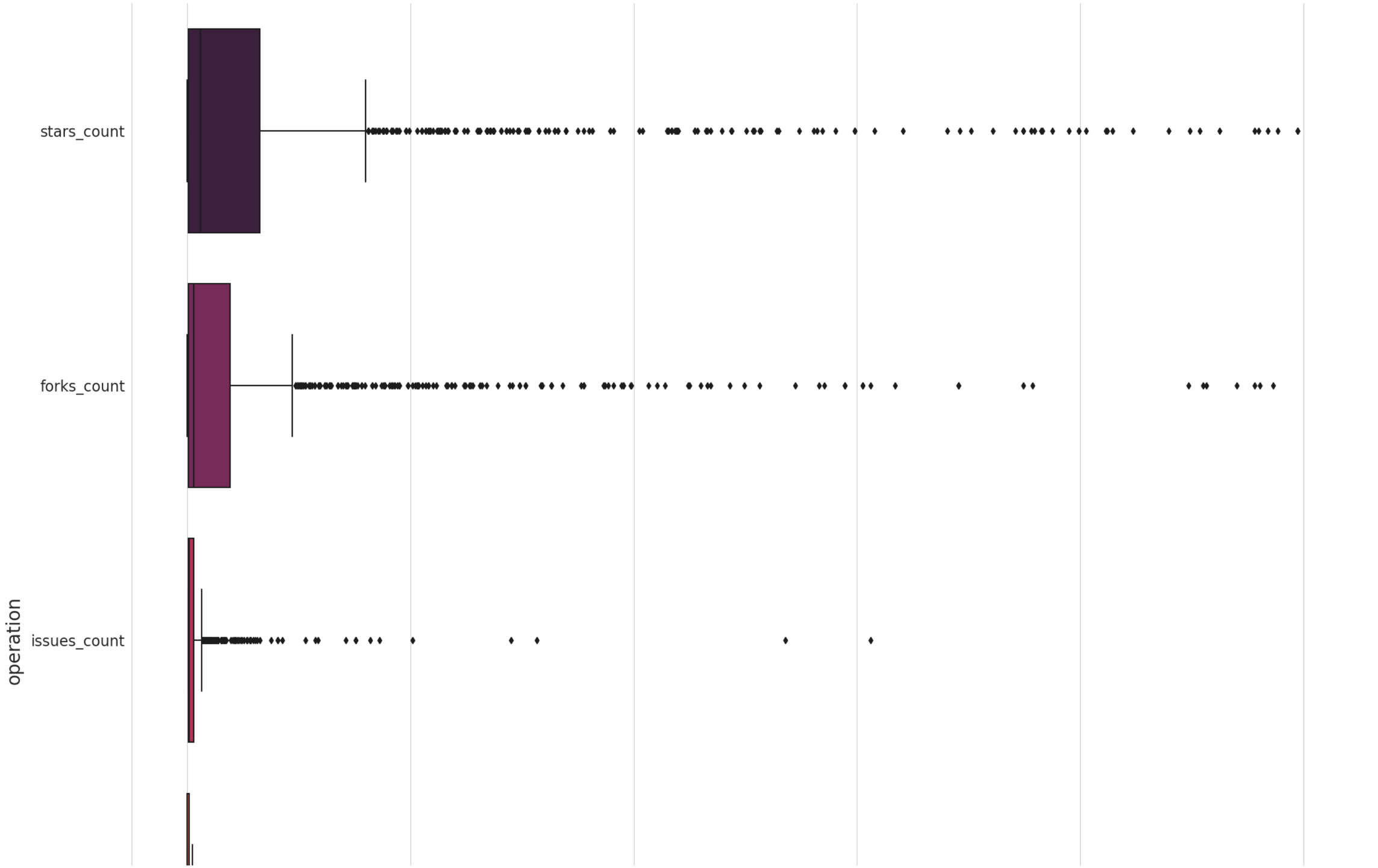
Out[31]:

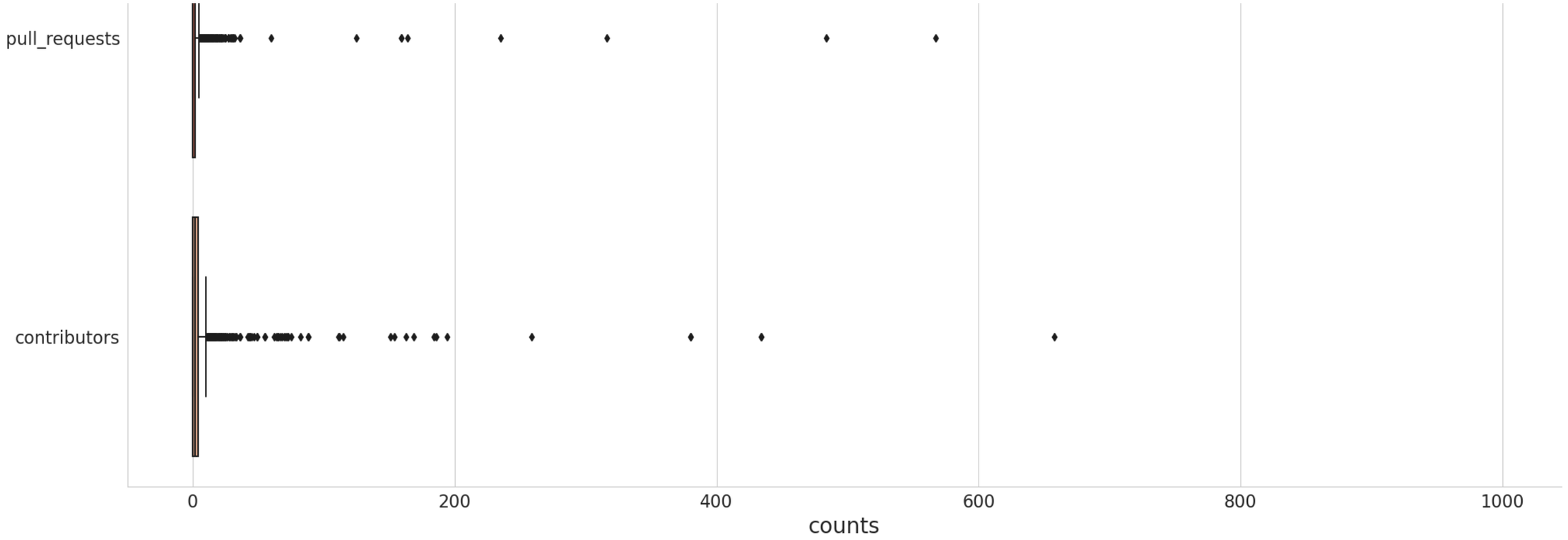
	operation	counts
0	stars_count	0
1	stars_count	271
2	stars_count	0
3	stars_count	0
4	stars_count	0
...	...	...
5255	contributors	0
5256	contributors	8
5257	contributors	7
5258	contributors	0
5259	contributors	0

5260 rows × 2 columns

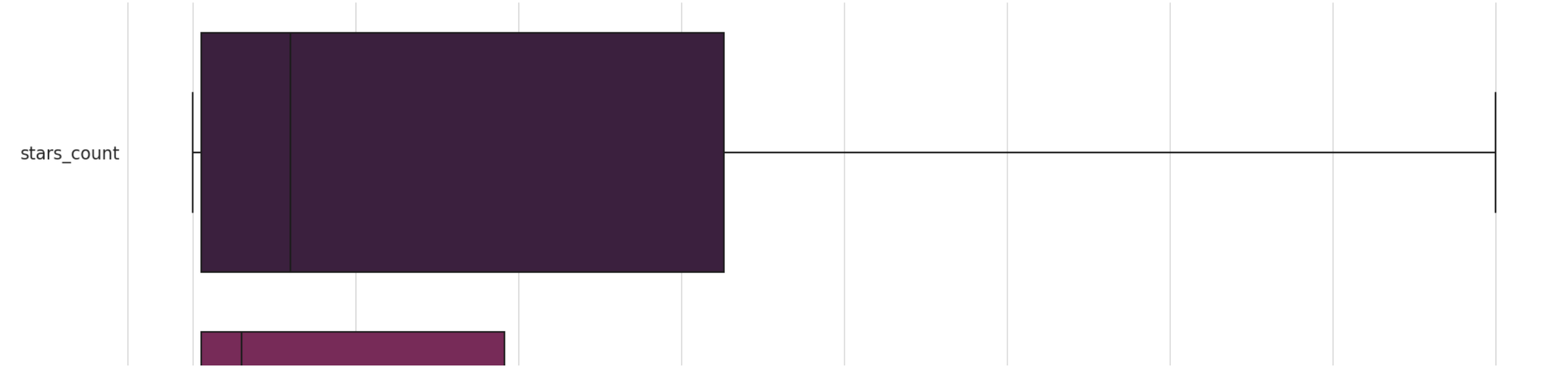
In [32]:
```
sns.set_style("whitegrid")
stars_count_box = sns.catplot(data=operation_counts, kind='box', y='operation', x='counts', height=20, palette='rocket')
plt.yticks(fontsize=16)
```

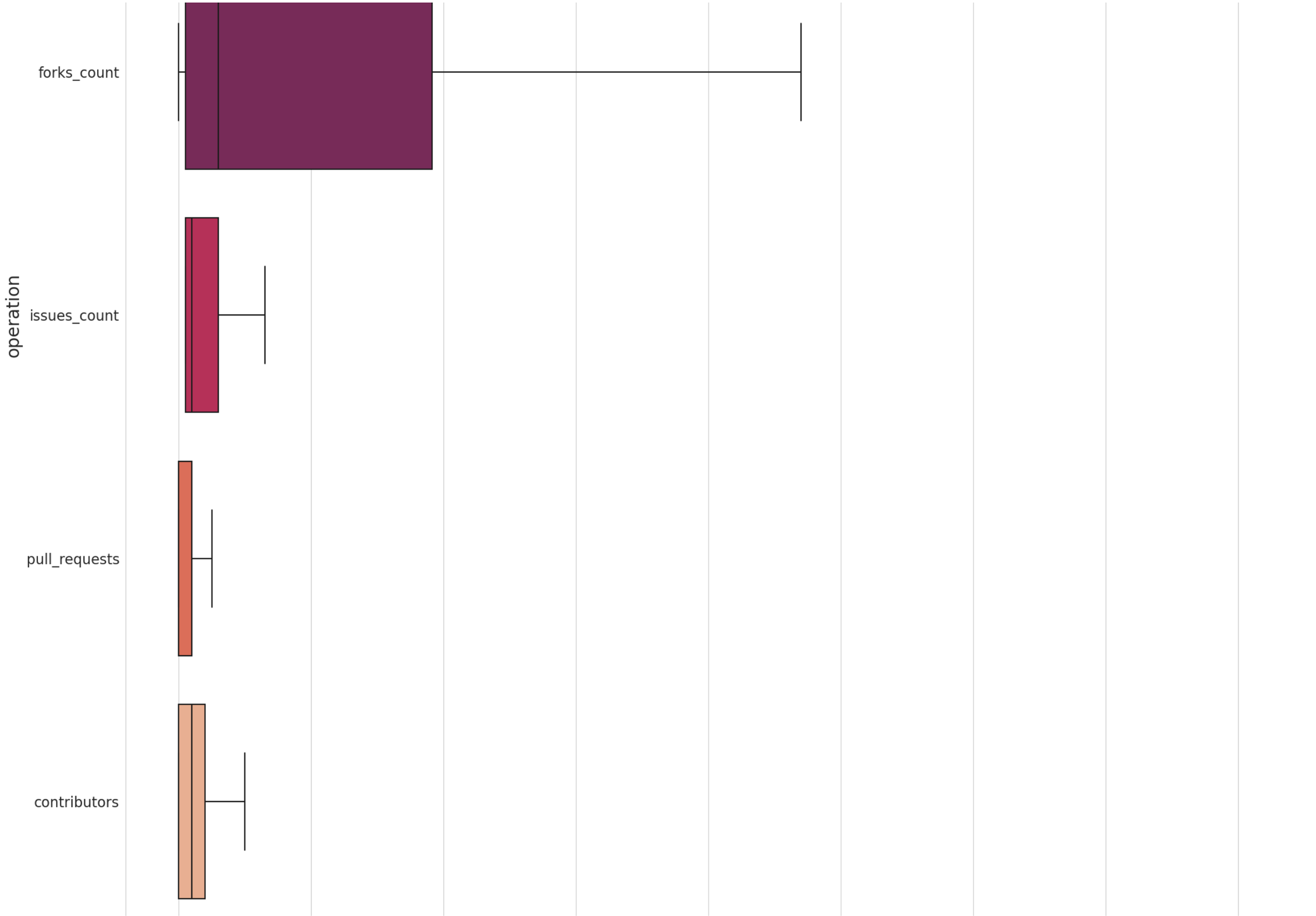
```
plt.xticks(fontsize=16)
plt.ylabel('operation', fontsize=20, )
plt.xlabel('counts', fontsize=20)
plt.show()
```

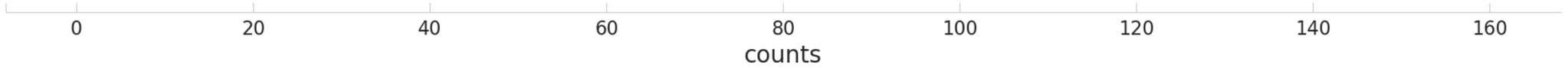




In [33]:
```
sns.set_style("whitegrid")
# No outliers
stars_count_box_no_outliers = sns.catplot(data=operation_counts, kind='box', y='operation', x='counts', height=20, palette='rocket', sym='')
plt.yticks(fontsize=16)
plt.xticks(fontsize=16)
plt.ylabel('operation', fontsize=20, )
plt.xlabel('counts', fontsize=20)
plt.show()
```




forks_count
operation
issues_count
pull_requests
contributors



3. Dealing With NaN

NaN Analysis

Noticed that there are NaN only in the column `language`. Github will recognize the programming language of the project automatically. When there is not a main program, and the repository is constructed by non-code documents, such as `.md`, then the information for `language` will possibly be NaN. Some repositories, for example *EddieHubCommunity/support*, have a lot of stars/forks/issues/pull requests/contributors, but the language is empty. There is no contradiction in these situations, because these repositories are good guidance with excellent `readme.md` files.

Delete NaN

The dataset cleaned in this way is named to be ***github_data_cleaned***.

In [34]:
```
from numpy import nan as NA
```

In [35]:
```
github_data.isnull()
```

Out[35]:

	repositories	stars_count	forks_count	issues_count	pull_requests	contributors	language
0	False	False	False	False	False	False	True
1	False	False	False	False	False	False	True
2	False	False	False	False	False	False	False
3	False	False	False	False	False	False	False
4	False	False	False	False	False	False	False
...	...	...	...	...	...	...	...
1047	False	False	False	False	False	False	False
1048	False	False	False	False	False	False	False
1049	False	False	False	False	False	False	False
1050	False	False	False	False	False	False	False
1051	False	False	False	False	False	False	False

1052 rows × 7 columns

In [36]:
```
# Delete rows containing NaN
github_data_cleaned = github_data.dropna()
github_data_cleaned
```

Out[36]:

	repositories	stars_count	forks_count	issues_count	pull_requests	contributors	language
2	ethereum/aleth	0	0	313	27	154	C++
3	localstack/localstack	0	0	290	30	434	Python
4	education/classroom	0	589	202	22	67	Ruby
5	shobhit97/open-gpstracker	0	0	172	0	3	Java
6	donnemartin/system-design-primer	0	0	164	164	115	Python
...	...	...	...	...	...	...	...
1047	Tyriar/canvas-astar.dart	2	1	1	0	0	Dart
1048	ankitkumar70777/github-slideshow	0	0	1	0	8	HTML
1049	aitikgupta/interactive_cpu_scheduler	0	5	1	1	7	Python
1050	gwmccubbin/voting_dapp	11	5	1	0	0	JavaScript
1051	gamemann/All_PropHealth	4	3	1	0	0	SourcePawn

907 rows × 7 columns

Replace NaN with the Value of the Highest Frequency

The dataset cleaned in this way is named to be ***github_data_HF_replaced***.

In [37]:
```
language_HF = language_counts.index[-1]
print('{} is the language with the highest frequency.'.format(language_HF))
```

```
JavaScript is the language with the highest frequency.
```

In [38]:
```
github_data_HF_replaced = github_data.replace(np.nan, language_HF)
github_data_HF_replaced
```

Out[38]:

	repositories	stars_count	forks_count	issues_count	pull_requests	contributors	language
0	octocat/Hello-World	0	0	612	316	2	JavaScript
1	EddieHubCommunity/support	271	150	536	6	71	JavaScript
2	ethereum/aleth	0	0	313	27	154	C++
3	localstack/localstack	0	0	290	30	434	Python
4	education/classroom	0	589	202	22	67	Ruby
...	...	...	...	...	...	...	...
1047	Tyriar/canvas-astar.dart	2	1	1	0	0	Dart
1048	ankitkumar70777/github-slideshow	0	0	1	0	8	HTML
1049	aitikgupta/interactive_cpu_scheduler	0	5	1	1	7	Python
1050	gwmccubbin/voting_dapp	11	5	1	0	0	JavaScript
1051	gamemann/All_PropHealth	4	3	1	0	0	SourcePawn

1052 rows × 7 columns

According to the raw dataset, the missing data is replaced by 'JavaScript', and we now construct a new cleaned dataset ***github_data_HF_replaced***.

Complement NaN Regarding to the Correlationships Between Attributes

The dataset cleaned in this way is named to be ***github_data_attr_corr***.

In [39]:
```
df_coded = pd.get_dummies(github_data, columns=['language'], dummy_na=True, drop_first=True)
df_coded.head()
```

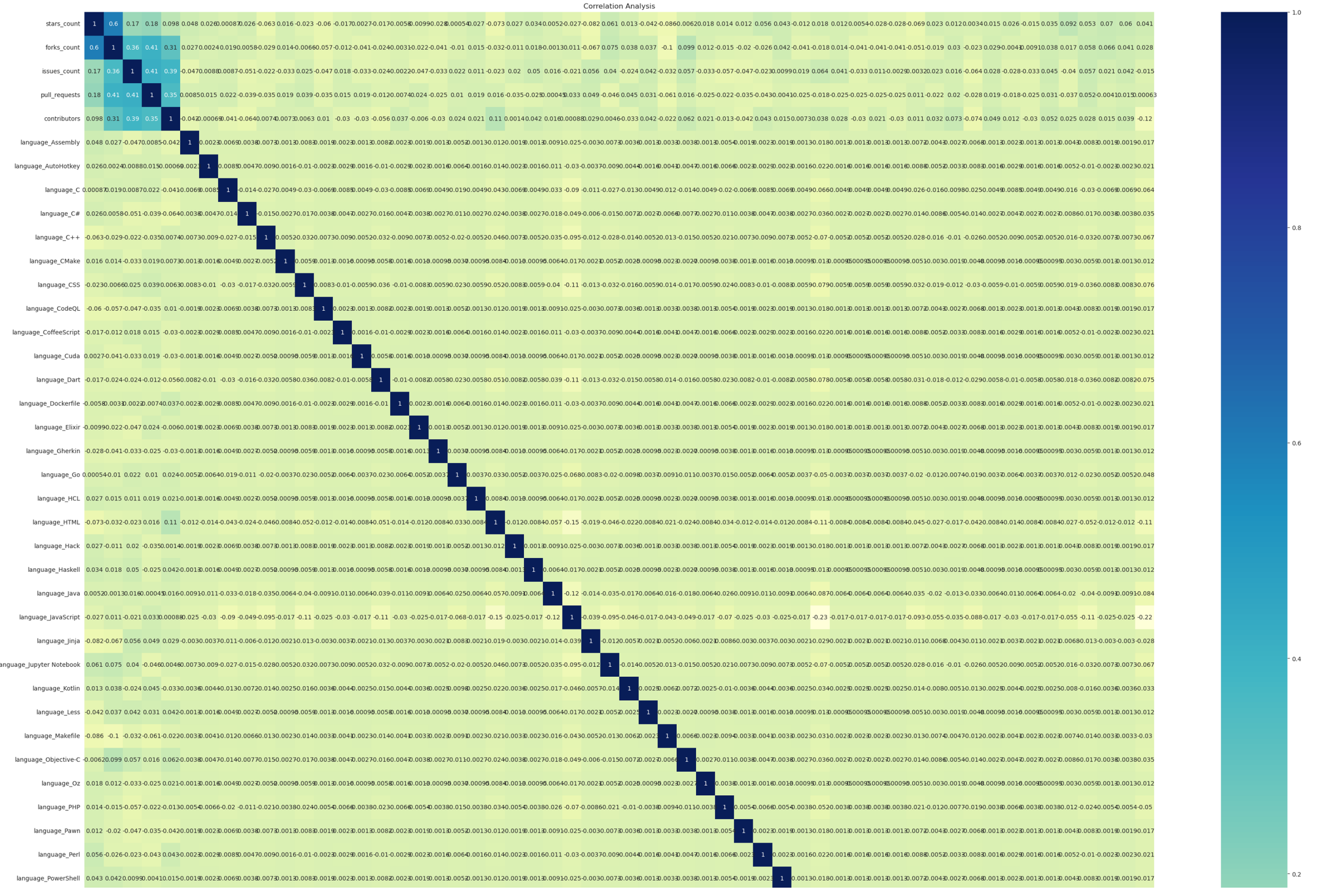
Out[39]:

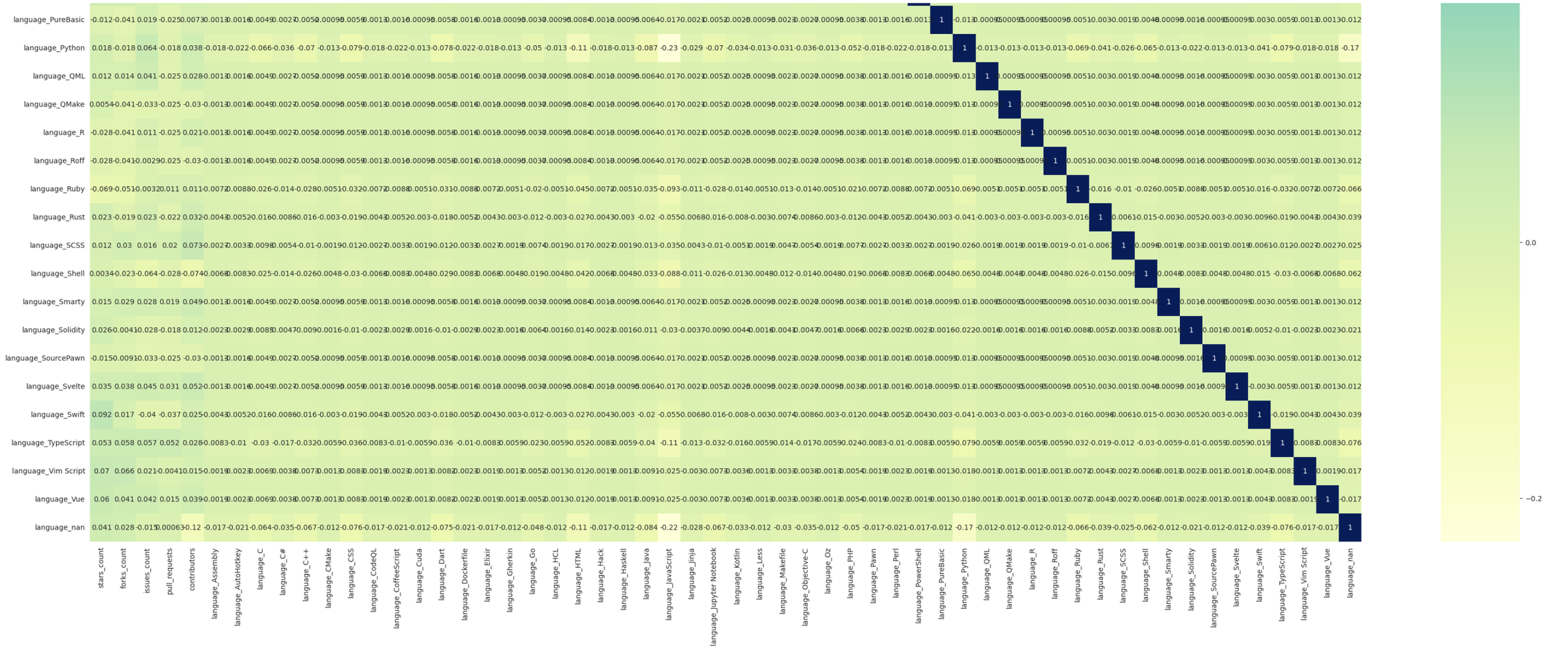
	repositories	stars_count	forks_count	issues_count	pull_requests	contributors	language_Assembly	language_AutoHotkey	language_C	language_C#	...	language_Shell	language_Smarty	langua
0	octocat/Hello-World	0	0	612	316	2	0	0	0	0	...	0	0	
1	EddieHubCommunity/support	271	150	536	6	71	0	0	0	0	...	0	0	
2	ethereum/aleth	0	0	313	27	154	0	0	0	0	...	0	0	
3	localstack/localstack	0	0	290	30	434	0	0	0	0	...	0	0	
4	education/classroom	0	589	202	22	67	0	0	0	0	...	0	0	

5 rows × 57 columns

In [40]:
```
plt.figure(figsize=(40, 40))
sns.heatmap(df_coded.corr(method='spearman'), cmap='YlGnBu', annot=True)
plt.title('Correlation Analysis')
```

Out[40]: Text(0.5, 1.0, 'Correlation Analysis')





The correlationship between the language used in a repository and the counts (star/fork/issue/pull/contributor) is so weak, thus use correlationship to complete the missing data is not a good idea.

In [41]:
```
github_data_attr_corr = github_data
```

In [42]:
```
df1 = github_data_attr_corr.groupby('language').agg(avg=('stars_count', 'mean'))
df1
```

Out[42]:

language	avg
ActionScript	3.000000
Assembly	426.000000
AutoHotkey	118.333333
C	153.076923
C#	43.250000
C++	39.172414

CMake	30.000000
CSS	49.675676
CodeQL	0.000000
CoffeeScript	7.333333
Cuda	14.000000
Dart	33.944444
Dockerfile	61.666667
Elixir	36.000000
Gherkin	1.000000
Go	139.600000
HCL	65.000000
HTML	54.888889
Hack	59.500000
Haskell	126.000000
Java	83.204545
JavaScript	87.083004
Jinja	0.400000
Jupyter Notebook	130.724138
Kotlin	26.714286
Less	0.000000
Makefile	0.833333
Objective-C	234.375000
Oz	37.000000
PHP	86.562500
Pawn	19.000000
Perl	294.000000
PowerShell	127.500000
PureBasic	5.000000
Python	73.038710
QML	24.000000
QMake	16.000000
R	1.000000
Roff	1.000000
Ruby	25.071429

Rust	56.400000
SCSS	228.500000
Shell	28.600000
Smarty	28.000000
Solidity	113.333333
SourcePawn	4.000000
Svelte	133.000000
Swift	152.500000
TypeScript	70.837838
Vim Script	602.000000
Vue	424.500000

In [43]:
```
for i in range(len(github_data_attr_corr)):
    if github_data_attr_corr['language'].iloc[i] is NA:
        rate = github_data_attr_corr['stars_count'].iloc[i]
        dist = []
        for j in range(len(df1)):
            dist.append(abs(df1.iloc[j]['avg']-rate))
        idx = dist.index(min(dist))
        github_data_attr_corr['language'].iloc[i] = df1.index[idx]
github_data_attr_corr['language'].value_counts()
```

```
/opt/conda/lib/python3.7/site-packages/pandas/core/indexing.py:1732: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
(https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy)
  self._setitem_single_block(indexer, value, name)
```

Out[43]:
```
JavaScript          257
Python              155
HTML                 72
Java                 44
CSS                  42
Dart                 38
TypeScript           38
C                    34
CodeQL               33
C++                  33
Jupyter Notebook     29
Ruby                 28
Shell                25
PHP                  17
Go                   17
Perl                 14
ActionScript         12
Swift                10
```

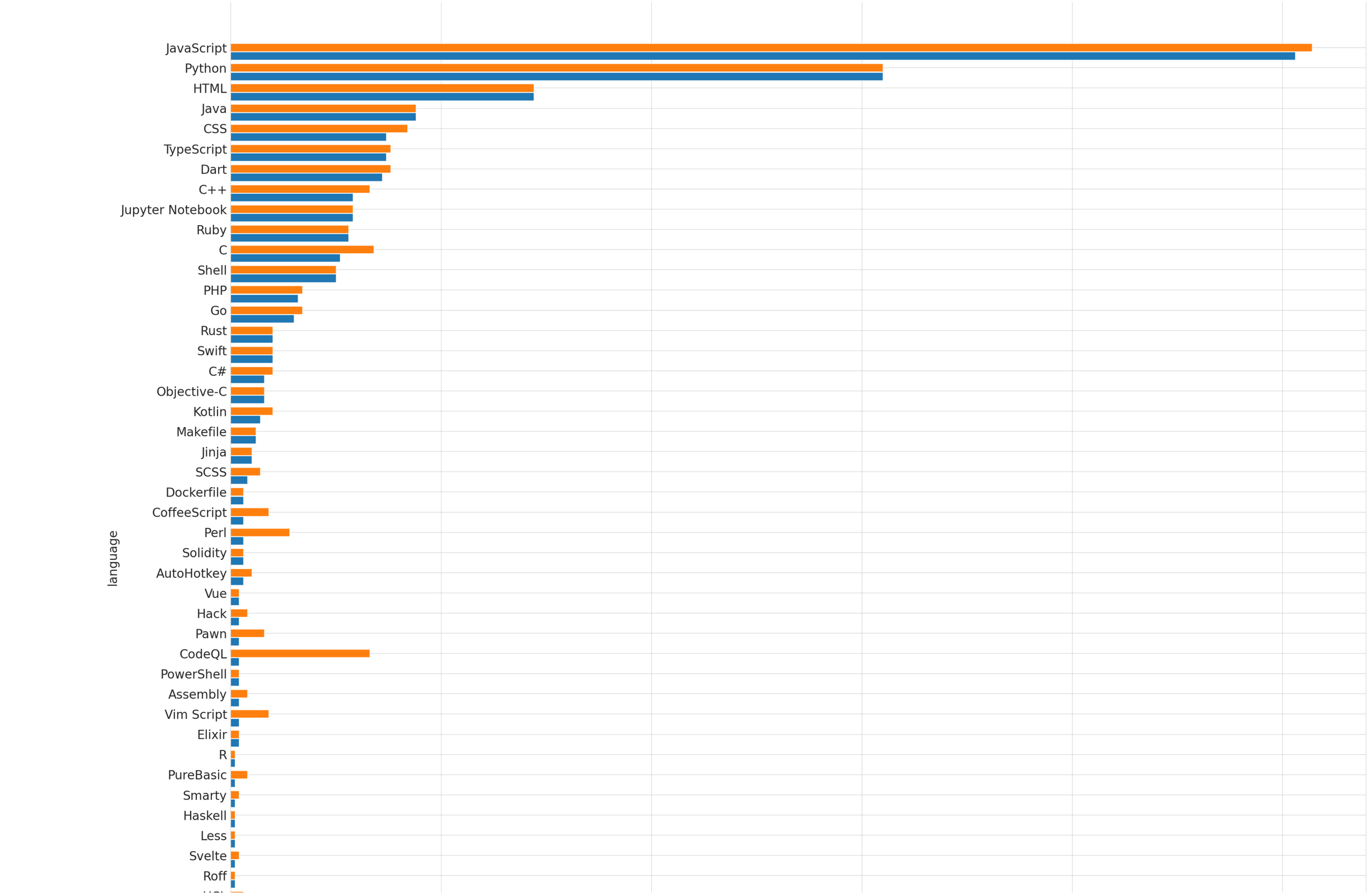
```
Rust                10
C#                  10
Kotlin              10
Vim Script           9
CoffeeScript         9
Pawn                 8
Objective-C          8
Cuda                 7
Gherkin              7
SCSS                 7
Makefile             6
SourcePawn           6
AutoHotkey           5
QMake                5
Jinja                5
Assembly             4
PureBasic            4
Hack                 4
QML                  3
CMake                3
Dockerfile           3
HCL                  3
Solidity             3
Elixir               2
Svelte               2
Smarty               2
Vue                  2
PowerShell           2
R                    1
Less                 1
Roff                 1
Haskell              1
Oz                   1
Name: language, dtype: int64
```

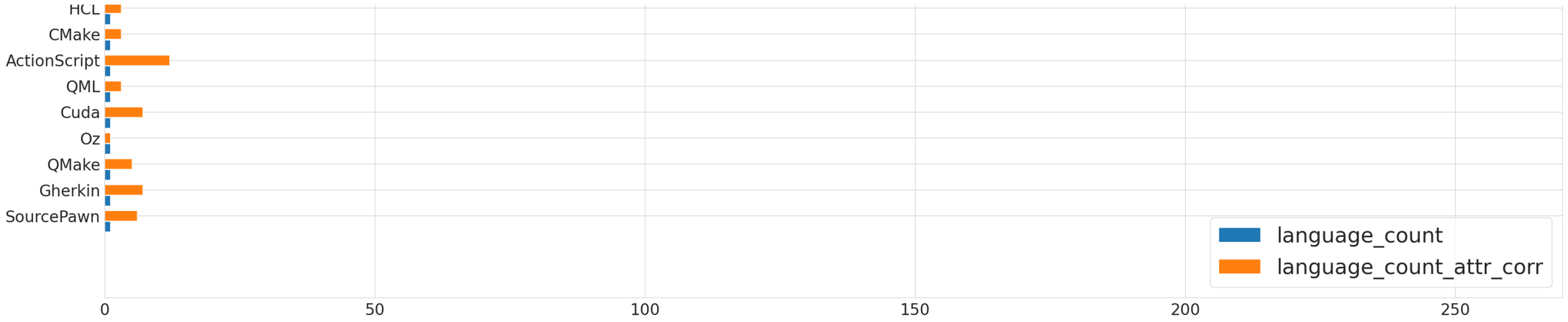
In [44]:

```
language_count_attr_corr = language_counts
language_count_attr_corr['language_count_attr_corr'] = [0] * len(language_counts)

for level in list(language_counts.index):
    if level in list(github_data_attr_corr['language'].value_counts().index):
        language_count_attr_corr.loc[[level], ['language_count_attr_corr']] = github_data_attr_corr['language'].value_counts().loc[[level]].values[0]

plt.figure(figsize=(40, 40))
plt.yticks(fontsize=24)
plt.xticks(fontsize=24)
plt.barh(list(range(len(language_count_attr_corr))), tick_label=language_count_attr_corr.index, width=language_count_attr_corr['language_count'], label='lar
plt.barh([d+0.42 for d in list(range(len(language_count_attr_corr)))], tick_label=language_count_attr_corr.index, width=language_count_attr_corr['language_c
plt.ylabel('language', fontsize=24)
plt.xlabel('', fontsize=24)
# plt.title('Number of movies for each appropriation-level?', fontsize=32, loc='center')
plt.legend(fontsize=32, loc='lower right')
plt.show()
```


JavaScript
Python
HTML
Java
CSS
TypeScript
Dart
C++
Jupyter Notebook
Ruby
C
Shell
PHP
Go
Rust
Swift
C#
Objective-C
Kotlin
Makefile
Jinja
SCSS
Dockerfile
CoffeeScript
Perl
Solidity
AutoHotkey
Vue
Hack
Pawn
CodeQL
PowerShell
Assembly
Vim Script
Elixir
R
PureBasic
Smarty
Haskell
Less
Svelte
Roff
language



This result shows that this method smooth the original distribution.

Substitute NaN Regarding to the Correlationships Between Samples

The dataset cleaned in this way is named to be ***github_data_sample_corr***.

In [45]:
```
github_data_sample_corr = github_data
```

In [46]:
```
def regularit(df):
    new_df = pd.DataFrame(index=df.index)
    columns = ['stars_count', 'forks_count', 'issues_count', 'pull_requests', 'contributors']
    for c in columns:
        d = df[c]
        MAX = d.max()
        MIN = d.min()
        new_df[c] = ((d - MIN) / (d - MAX))
    return new_df
```

In [47]:
```
normal_github_data = regularit(github_data_sample_corr)
normal_github_data
```

Out[47]:

	stars_count	forks_count	issues_count	pull_requests	contributors
0	-0.000000	-0.000000	inf	-1.258964	-0.003049
1	-0.374309	-0.182260	-7.039474	-0.010695	-0.120954
2	-0.000000	-0.000000	-1.043478	-0.050000	-0.305556
3	-0.000000	-0.000000	-0.897516	-0.055866	-1.937500
4	-0.000000	-1.533854	-0.490244	-0.040367	-0.113367
...	...	...	...	...	...
1047	-0.002014	-0.001029	-0.000000	-0.000000	-0.000000
1048	-0.000000	-0.000000	-0.000000	-0.000000	-0.012308
1049	-0.000000	-0.005165	-0.000000	-0.001767	-0.010753
1050	-0.011179	-0.005165	-0.000000	-0.000000	-0.000000
1051	-0.004036	-0.003093	-0.000000	-0.000000	-0.000000

1052 rows × 5 columns

In [48]:
```
normal_language = pd.concat([normal_github_data, github_data_sample_corr['language']], axis=1)
normal_language
```

Out[48]:

	stars_count	forks_count	issues_count	pull_requests	contributors	language
0	-0.000000	-0.000000	inf	-1.258964	-0.003049	CodeQL
1	-0.374309	-0.182260	-7.039474	-0.010695	-0.120954	Perl
2	-0.000000	-0.000000	-1.043478	-0.050000	-0.305556	C++
3	-0.000000	-0.000000	-0.897516	-0.055866	-1.937500	Python
4	-0.000000	-1.533854	-0.490244	-0.040367	-0.113367	Ruby
...	...	...	...	...	...	...
1047	-0.002014	-0.001029	-0.000000	-0.000000	-0.000000	Dart
1048	-0.000000	-0.000000	-0.000000	-0.000000	-0.012308	HTML
1049	-0.000000	-0.005165	-0.000000	-0.001767	-0.010753	Python
1050	-0.011179	-0.005165	-0.000000	-0.000000	-0.000000	JavaScript
1051	-0.004036	-0.003093	-0.000000	-0.000000	-0.000000	SourcePawn

1052 rows × 6 columns

In [49]:
```
infos = []
for i in range(len(normal_language)):
    info = []
    star = normal_language['stars_count'].iloc[i]
    fork = normal_language['forks_count'].iloc[i]
    issue = normal_language['issues_count'].iloc[i]
    pull = normal_language['pull_requests'].iloc[i]
    contributor = normal_language['contributors'].iloc[i]
    info.append(star)
    info.append(fork)
    info.append(issue)
    info.append(pull)
    info.append(contributor)
    infos.append(info)
```

In [50]:
```
for i in range(len(normal_language)):
    if normal_language['language'].iloc[i] is NA:
        dists = []
        for j in len(rates):
            dist = np.sqrt(np.sum(np.square(infos[i] - infos[j])))
            dists.append(dist)
        idx = dists.index(min(dists))
        github_data_sample_corr['language'].iloc[i] = github_data_sample_corr['language'].iloc[idx]
github_data_sample_corr['language'].value_counts()
```

Out[50]:
```
JavaScript          257
Python              155
HTML                 72
Java                 44
CSS                  42
Dart                 38
TypeScript           38
C                    34
CodeQL               33
C++                  33
Jupyter Notebook     29
Ruby                 28
Shell                25
PHP                  17
Go                   17
Perl                 14
ActionScript         12
Swift                10
Rust                 10
C#                   10
Kotlin               10
Vim Script            9
CoffeeScript          9
Pawn                  8
Objective-C           8
Cuda                  7
Gherkin               7
SCSS                  7
```

```
Makefile             6
SourcePawn           6
AutoHotkey           5
QMake                5
Jinja                5
Assembly             4
PureBasic            4
Hack                 4
QML                  3
CMake                3
Dockerfile           3
HCL                  3
Solidity             3
Elixir               2
Svelte               2
Smarty               2
Vue                  2
PowerShell           2
R                    1
Less                 1
Roff                 1
Haskell              1
Oz                   1
Name: language, dtype: int64
```

This result is quite similar to the previous method. The reason is probably that the count attributes we use here are highly correlated, so that when we use them to measure the correlationships between samples, the result does not change much.