Tool Support for Commenting Conventions

Bachelor’s Thesis – Final Presentation, 29/06/2021

By Michael Dooley

Supervised by Pooja Rani and Nataliia Stulova

University of Bern

Comments

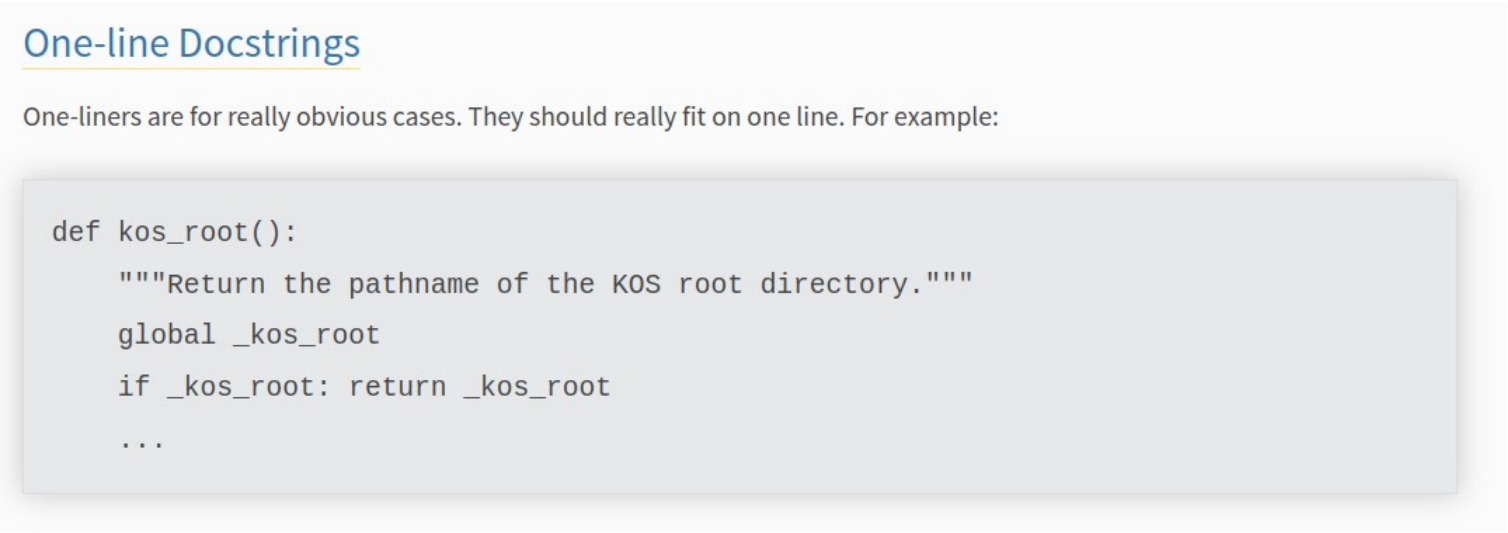
- Comments help developers
- Good Comments follow guidelines

```
/**
* Returns an Image object that can then be painted on the screen.
* The url argument must specify an absolute <a href="#{@link}">{@link URL}</a>. The name
* argument is a specifier that is relative to the url argument.
* <p>
* This method always returns immediately, whether or not the
* image exists. When this applet attempts to draw the image on
* the screen, the data will be loaded. The graphics primitives
* that draw the image will incrementally paint on the screen.
*
* @param  url  an absolute URL giving the base location of the image
* @param  name the location of the image, relative to the url argument
* @return      the image at the specified URL
* @see         Image
*/
```

(Example from https://www.oracle.com/technical-resources/articles/java/javadoc-tool.html)

Style Guidelines

- Codification of best stylistic practices
- For comments: Cover both content and style

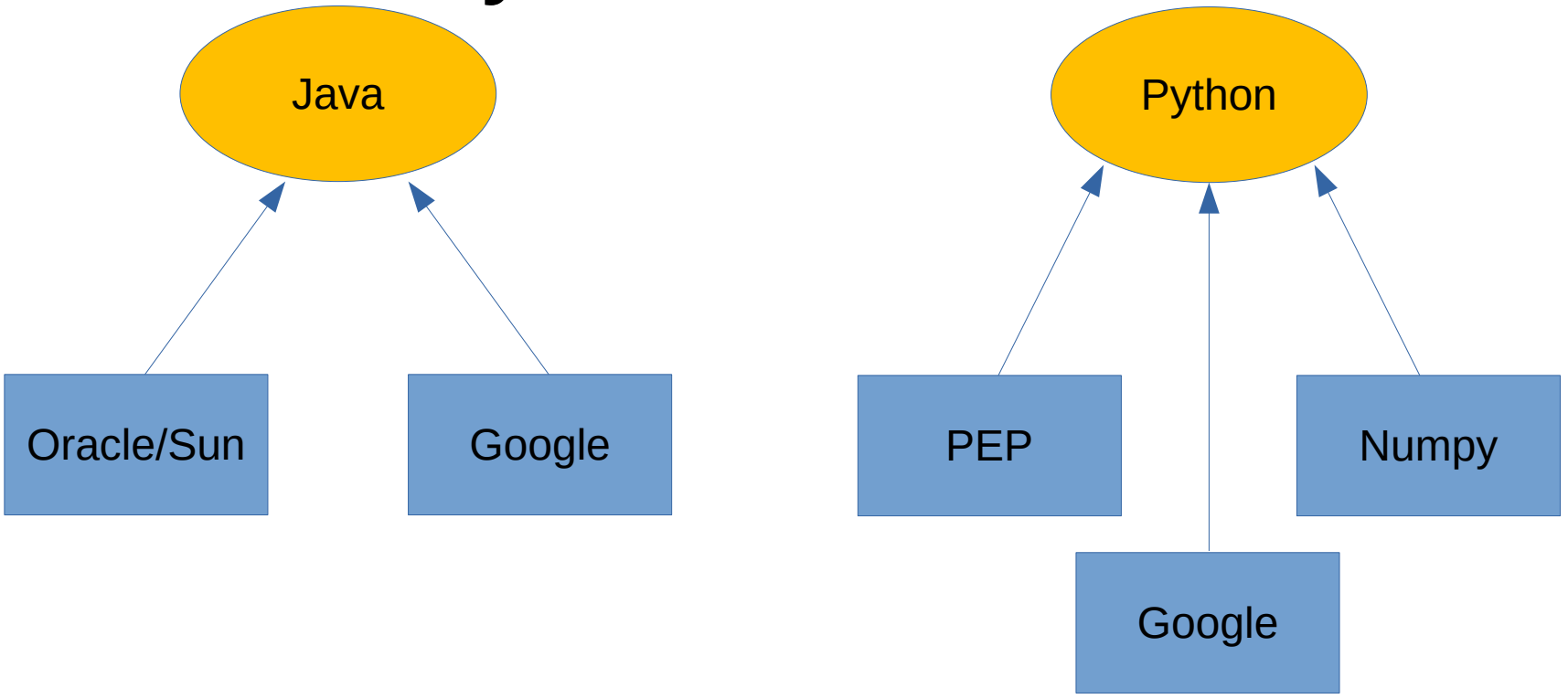


One-line Docstrings

One-liners are for really obvious cases. They should really fit on one line. For example:

```
def kos_root():
    """Return the pathname of the KOS root directory."""
    global _kos_root
    if _kos_root: return _kos_root
    ...
```

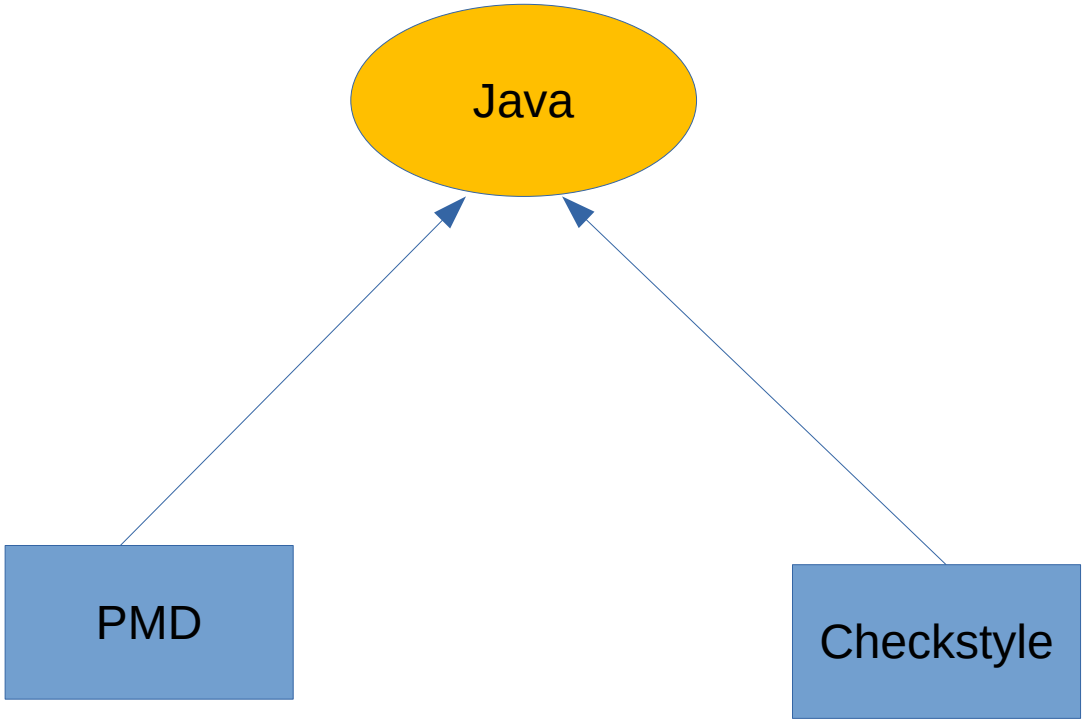
Style Guidelines



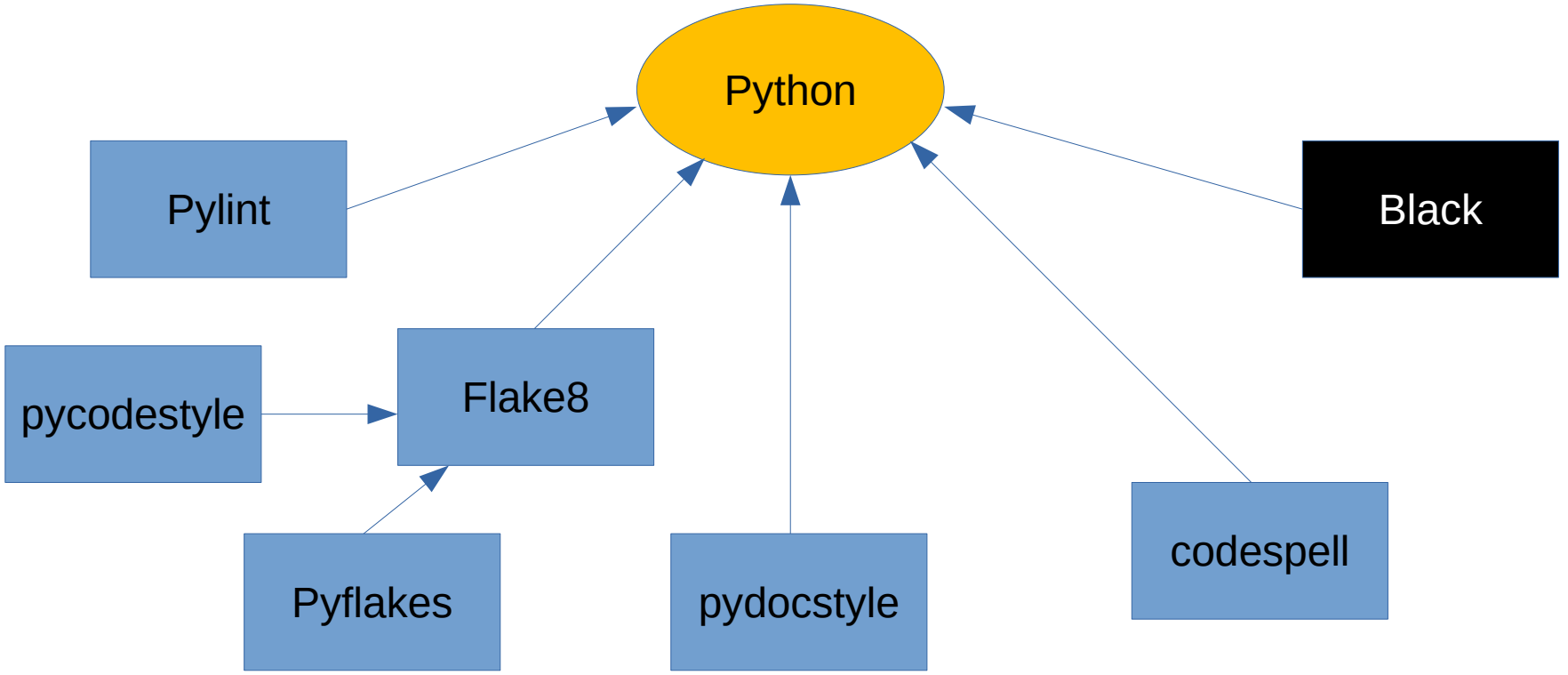
Style Checkers

Programs, that check code for consistent style

Style Checkers



Style Checkers



Motivation

- Style Guidelines cover comments
- Style Checkers check comments

...but how?

Question 1

How do suggestions for comments from style guidelines differ from checks by style checkers?

Related: How do style checkers differ compared to each other?

Approach

- Find all rules, see what matches.

Coverage – Python

Coverage	Style Checkers							
Style Guidelines		pydocstyle	Flake8	pycodestyle	Pylint	Black	codespell	Pyflakes
	PEP	24%	15%	15%	7%	6%	0%	0%
	Google	17%	8%	8%	7%	1%	1%	0%
	Numpy	4%	1%	1%	2%	0%	0%	0%

Coverage – Java

Coverage	Style Checkers		
Style Guidelines		Checkstyle	PMD
	Oracle/Sun	14%	3%
	Google	80%	20%

Rules not in Guidelines

Python						
pydocstyle	Flake8	Pyflakes	pycodestyle	Pylint	Black	codespell
26	3	2	1	1	0	0

Java	
Checkstyle	PMD
42	6

Examples

- No empty Docstrings (Pylint)
- Only document existing params (Checkstyle)
- Document enums (Checkstyle & PMD)

What types of rules are covered?

Coverage	Formatting	Writing Style	Structure	Syntax	Content
Python	42%	28%	26%	13%	3%
Java	23%	0%	33%	15%	19%

Findings

- Coverage not very high
- Results by type differ across language

Question 2

How are style checkers used for comments in practice?

Approach

- Collect 100 most-starred projects on Github per language
- Remove non-suitable projects (48 each remaining)
- Gather configuration data
- Run Style Checkers

What Style Chekers are used?

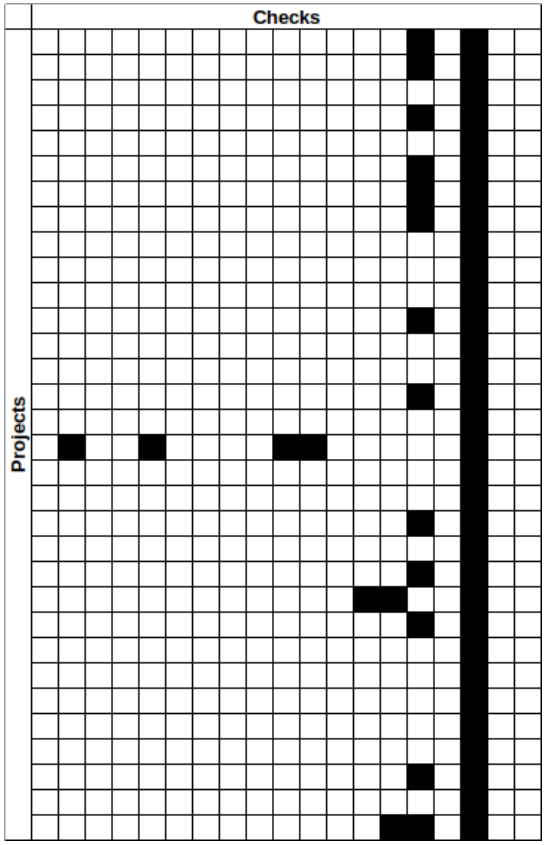
Python							
Flake8	Black	Pylint	pydocstyle	pycodestyle	codespell	Pyflakes	None
32	14	11	4	3	3	0	7

Java		
Checkstyle	PMD	None
26	9	21

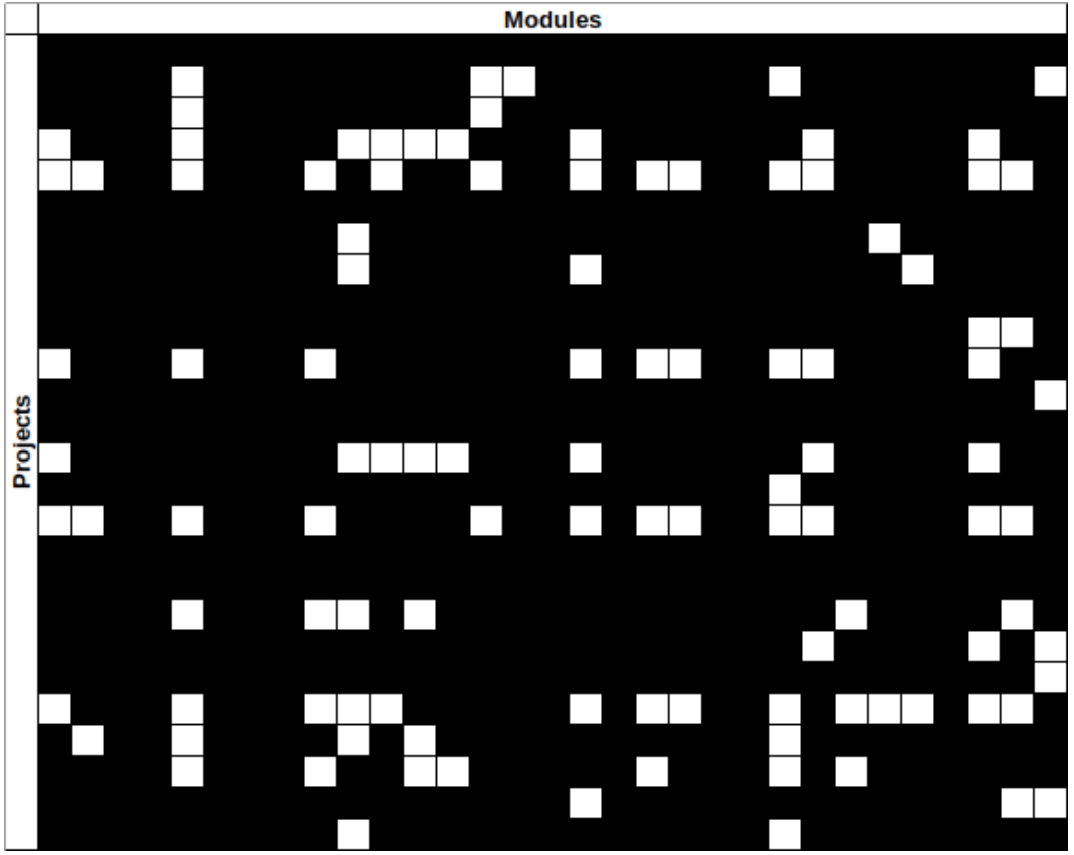
Configuration

- XML-Configs changed more commonly than command-line configs
- Line Length most commonly changed/disabeled
- 2 projects with invalid config

Active Checks – Flake8



Active Checks – Checkstyle



Style Checker Violations - Flake8

you-get	841	YouCompleteMe	3	faceswap	0	django-rest-framework	0
scikit-learn	187	core	1	fastapi	0	tornado	0
glances	51	celery	1	localstack	0	tqdm	0
scrapy	32	youtube-dl	0	pandas	0	hosts	0
sherlock	32	thefuck	0	sentry	0	detectron2	0
redash	18	Django	0	compose	0	locust	0
requests	5	Flask	0	mitmpoxy	0	ray	0
pipenv	4	transformers	0	airflow	0		

Style Checker Violations – Checkstyle

elasticsearch	48699	picasso	17	flink	1	dubbo	0
apollo	3181	disruptor	4	kafka	0	zxing	0
okhttp	417	seata	4	spring-cloud-alibaba	0	spring-boot	0
glide	382	tinker	4	jadx	0	redisson	0
nacos	102	rocketmq	4	RxJava	0	netty	0
shardingsphere	95	spring-framework	1	jenkins	0	NewPipe	0

Findings

- Best tools not always used
- Configuration either close to default or fully individual across projects
- Style checker being there doesn’t mean it’s used

Summary

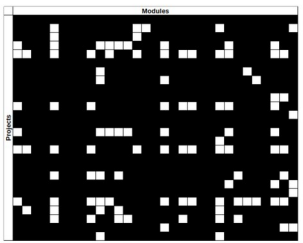
Coverage – Python

Coverage		Style Checkers						
		pydocstyle	Flake8	pycodestyle	Pylint	Black	codespell	Pyflakes
Style Guidelines	PEP	24%	15%	15%	7%	6%	0%	0%
	Google	17%	8%	8%	7%	1%	1%	0%
	Numpy	4%	1%	1%	2%	0%	0%	0%

What types of rules are covered?

	Formatting	Writing Style	Structure	Syntax	Content
Python	42%	28%	26%	13%	3%
Java	23%	0%	33%	15%	19%

Active Checks – Checkstyle



Style Checker Violations - Flake8

you-get	841	YouCompleteMe	3	faceswap	0	django-rest-framework	0
scikit-learn	187	core	1	fastapi	0	tornado	0
glances	51	celery	1	localstack	0	tqdm	0
scrapy	32	youtube-dl	0	pandas	0	hosts	0
sherlock	32	thefuck	0	sentry	0	detectron2	0
redash	18	Django	0	compose	0	locust	0
requests	5	Flask	0	mitmpoxy	0	ray	0
pipenv	4	transformers	0	airflow	0		