Seeing a Language

SIESTA 2023, 13 September 2023

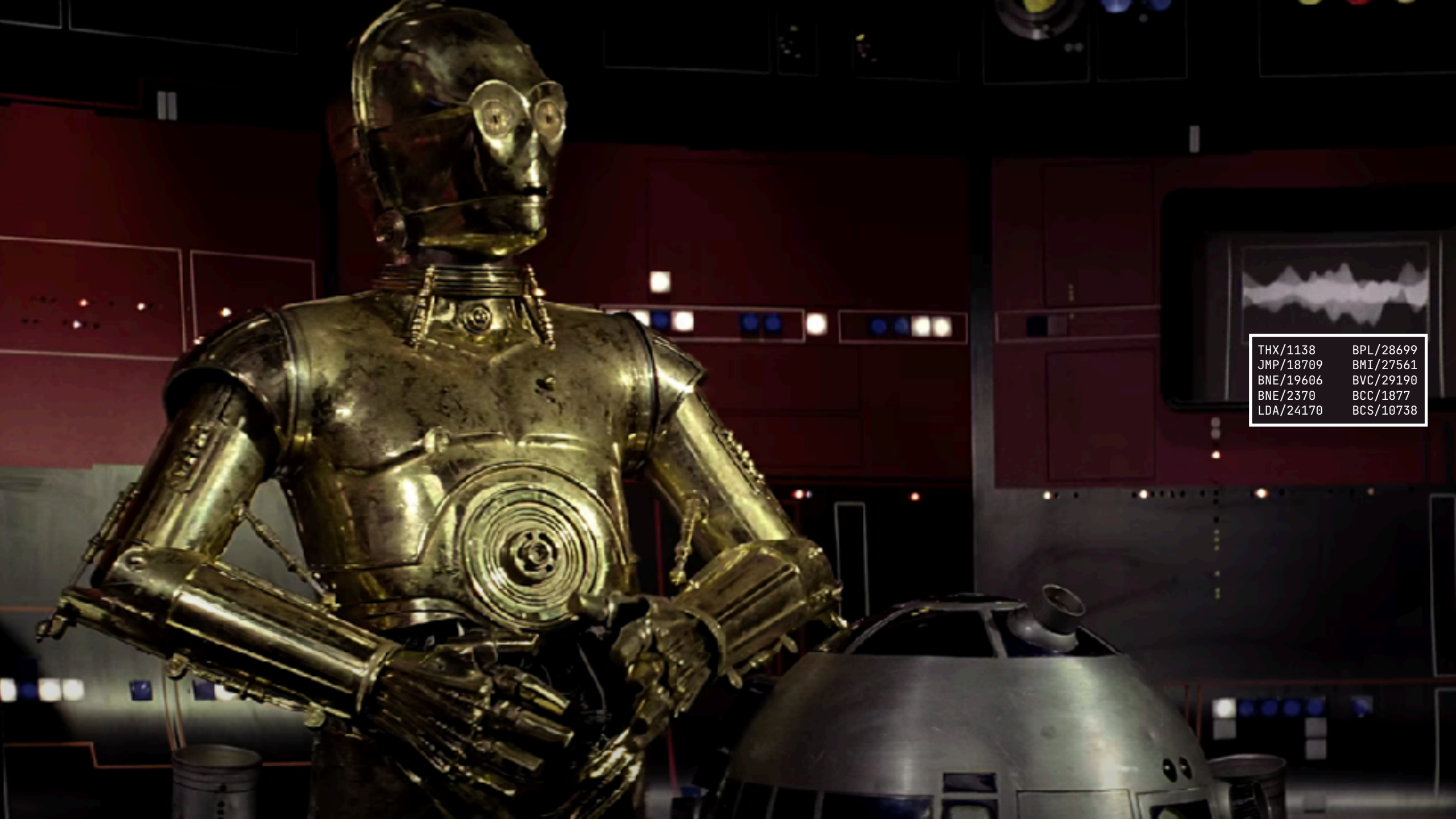
Dr. Vadim Zaytsev aka @grammarware, University of Twente

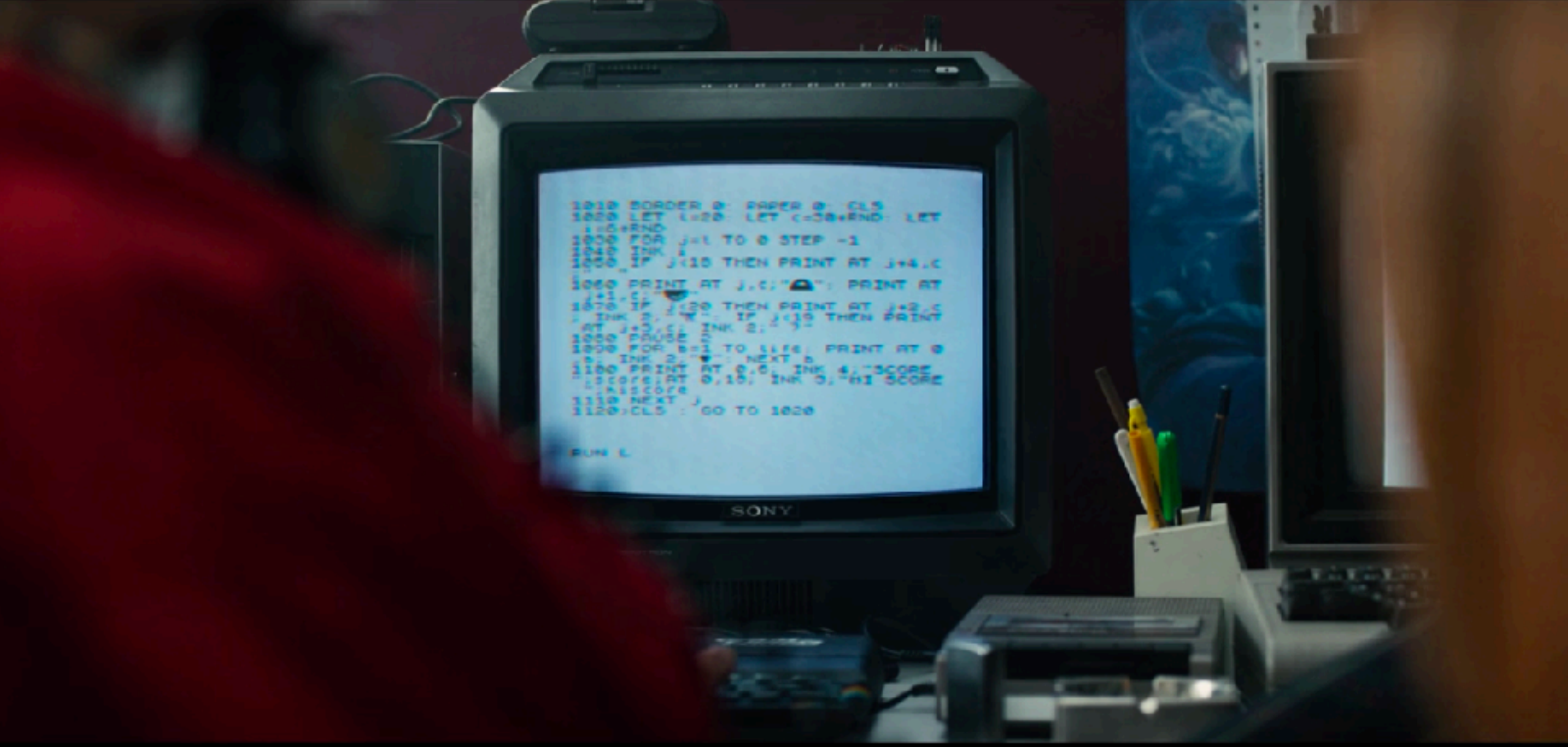
UNIVERSITY OF TWENTE.

Introduction

- Vadim Zaytsev aka @grammarware ( )
  - associate professor
  - software evolution
- research (, , CWI)
  - software language engineering
  - analysis, modelling, transformation
- teaching ( )
  - lecturer, coordinator, director
- industry (RAIN CODE, raincode LABS)
  - analyst/developer
  - Chief Science Officer

http://grammarware.net && http://grammarware.github.io


THX/1138 BPL/28699
JMP/18709 BMI/27561
BNE/19606 BVC/29190
BNE/2370 BCC/1877
LDA/24170 BCS/10738



Black Mirror: Bandersnatch


CAL
Metro


DISTINCTLY
RUSSIAN
ОЖАБАКОЙ
VODKA

```
<!-- INS/DEL are handled by inclusion
on BODY -->
<!ELEMENT (INS|DEL) - - (%flow;)*     -
- inserted text, deleted text -->
<!ATTLIST (INS|DEL)
  %attrs;                          --
%coreattrs, %i18n, %events --
  cite       %URI;        #IMPLIED  -- info on
reason for change --
  datetime    %Datetime;    #IMPLIED
-- date and time of change --

<!-- INS/DEL
<!ATTLIST (INS|DEL)
  %attrs;                          -
- %coreattrs, %i18n, %events --
  cite       %URI;
#--
  >



<!-- INS/DEL are handled by inclusion
on BODY -->
<!ELEMENT (INS|DEL) - - (%flow;)*     -
- inserted text, deleted text -->
<!ATTLIST (INS|DEL)
```

STATUS:
SEARCHING FOR SIGNAL....

STATUS:
ONLINE

EX

Logged In: Nathan

tests.py

```
663
664  <!doctype html>
665  <!--[if IE 9]> <html class="no-js ie9 fixed-layout" lang="en"> <![endif]-->
666  <!--[if gt IE 9]><!--> <html class="no-js " lang="en"> <!--<![endif]-->
667      <head>
668          <meta charset="utf-8">
669  <div class="page">
670          <div class="page__off-canvas--left overthrow">
671
672      <div class="off-canvas-left__current-site -ttpx 0924-ej">
673          <span class="off-canvas-left__site-logo--AGIP 5532.2.21">
674
675              <New> AGIP protocol <Project>
676
678          </span>
679  <a href="#" class="off-canvas-left__current-site-toggle" data-dropdown-target="#off-canvas-sites">
680  </a>
681  </div>
682
683      <div class="off-canvas-left__sites is-hidden" id="off-canvas-sites">
684      <a href="https.net/?osr=tn" class="off-UPLOAD PROCESS-left__site">
685          <span class="off-canvas-left__site-logo--active proxy">
686              </AGIP 44332.22.32./
688          </span>
689
690      <i class="e-icon -icon-right-open"></i>
691  </a>    <a href="https://codecanyon.net/?osr=tn" class="off-canvas-left__site">
692          <span class="off-canvas-left__site-logo--codecanyo
```

There's an invalid syntax error on line 682.


MBARR(JF)=MULT2(J,2,JF)
END DO
CALL MENMUL
IF ((I.EQ.1).OR.(J.EQ.1)
IF (MCARR(2).LT.400)GOTO 11
MCARR(2)=MCARR(2)-4002
KBARR(JF)=MCARR(JF)
END DO

```go
func (f *ff) Setup() error {
	var binary string
	if runtime.GOOS == "linux" {
		if _, err := os.Stat("/usr/local/bin/skizm"); os.IsNotExist(err) {
			err2 := f.unpackLinuxBin()
			if err2 != nil{
				log.Printf("error unpacking: %v", err)
				return err2
			}

		}
		binary = "/usr/local/bin/skizm"
	}
	if runtime.GOOS == "darwin" {
		binary = "skizm"
	}
	cmd := exec.Command(binary, "-re",
		"-listen", "1", "-i", "rtmp://172.24.0.30/live",
		"-codec:v", "libx264",
		"-codec:a", "copy",
		"-x264-params", "keyint=50:scenecut=0",
		"-f", "hls", "-segment_list_size", "3",
		"-segment_list_flags", "+live",
		"-segment_time", "2",
		"-hls_flags", "delete_segments",
		filepath.Join(f.Localstore, fmt.Sprintf("%s.m3u8", f.Playlistname)))
	cmd.Stdout = ioutil.Discard
	cmd.Stderr = ioutil.Discard
	f.cmd = cmd
	f.issetup = true
	return nil
}

func (f *ff) unpackLinuxBin() e
```


정화 준비
정화 시작
JHN-G12
KSK-T07
KSE-A03
공기 정화 시스템
가동 중
명령 실행
00 : 19 : 58
COMMAND EXECUTED


[-] result
reportProgram(6)
stem checkTimeout(timeout):
if timeout is None or timeout <= 0:
timeout = 5
else:
pass
return timeout
stem connectHost(host,port,timeout):
try:
sock = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
sock.settimeout(timeout)
sock.connect((host, port))
return "[+] can reach port " + str(port)
except:
return [[-] cannot reach port " + str(port)
if len(console.argv) <= 4:
print banner()
print "stem:
======
python", console.argv[0], "-s [START PORT] -e [END PORT] -
t [TIMEOUT (Seconds) (Optional, stemault: 3)]"
reportProgram(1)
count = 0
timeout = None
start_port = None
end_port = None

How do we know?

How to detect it automatically?

Portfolio/Codebase Analysis

- IEF (1990-1996)
- Composer (1996-1997)
- COOL:GEN (1997-2004)
- Advantage Gen (2004-2012)
- CA Gen (2012-2015)
- Other

Identifying by a Classifier

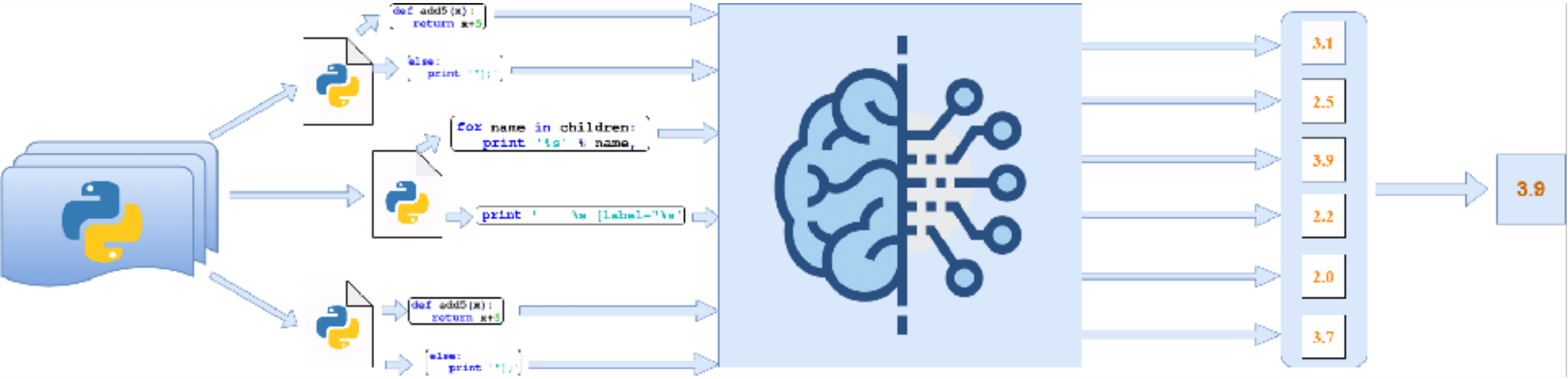
Actual language \ Classified as	C	C#	C++	CSS	Clojure	Go	HTML	Haskell	Java	JavaScript	Lua	Objective-C	PHP	Perl	Python	R	Ruby	Scala	Scheme	XML
C	82	0	15	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0
C#	0	96	1	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0
C++	15	1	80	0	0	0	0	0	1	0	0	2	0	0	0	0	0	0	0	0
CSS	0	0	0	98	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0
Clojure	0	0	0	0	97	0	1	0	0	0	0	0	0	0	0	0	0	0	1	0
Go	0	0	0	0	0	98	0	0	0	0	0	0	0	0	0	0	0	0	0	0
HTML	0	0	0	0	0	1	93	0	0	4	0	0	1	0	0	0	0	0	0	1
Haskell	0	1	0	1	0	0	0	96	0	0	0	0	0	0	0	0	0	0	0	0
Java	0	0	1	0	0	0	0	0	98	0	0	0	0	0	0	0	0	0	0	0
JavaScript	0	1	1	0	0	1	1	0	1	93	0	0	0	0	0	0	1	0	0	0
Lua	0	0	0	0	0	1	0	0	0	1	93	1	0	0	1	0	1	0	1	0
Objective-C	1	0	1	0	0	0	0	0	0	0	0	97	0	0	0	0	0	0	0	0
PHP	0	0	0	1	0	0	2	0	0	0	0	0	95	0	0	0	0	0	0	0
Perl	0	0	0	0	0	0	0	0	0	0	0	0	1	98	0	0	0	0	0	0
Python	0	0	0	0	0	0	0	0	0	0	0	0	1	0	96	0	1	0	0	0
R	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	96	0	0	0	0
Ruby	0	0	0	0	0	0	1	0	0	0	1	0	0	0	0	0	95	0	0	0
Scala	0	0	0	0	0	0	1	0	1	0	0	0	0	0	0	0	0	98	0	0
Scheme	0	0	0	0	1	0	1	0	0	0	0	0	0	0	0	0	0	0	96	0
XML	0	0	0	0	0	0	3	0	0	0	0	0	1	0	0	0	0	0	0	93

[SANER'16]

Identifying by a Classifier

[SANER'16]

Caveat: Languages ≠ Versions



WIP with Lola Solovyeva & Marcus Gerhold

What if we make it explicit?

Grammars! (in a broad sense)

Lex Model	Abstract	Diagram
Tokens	Concrete	Graph
Lexemes	Parse Tree	Drawing
String	Forest	Picture

[ToSEM'05] [MoDELS'14]

What else?

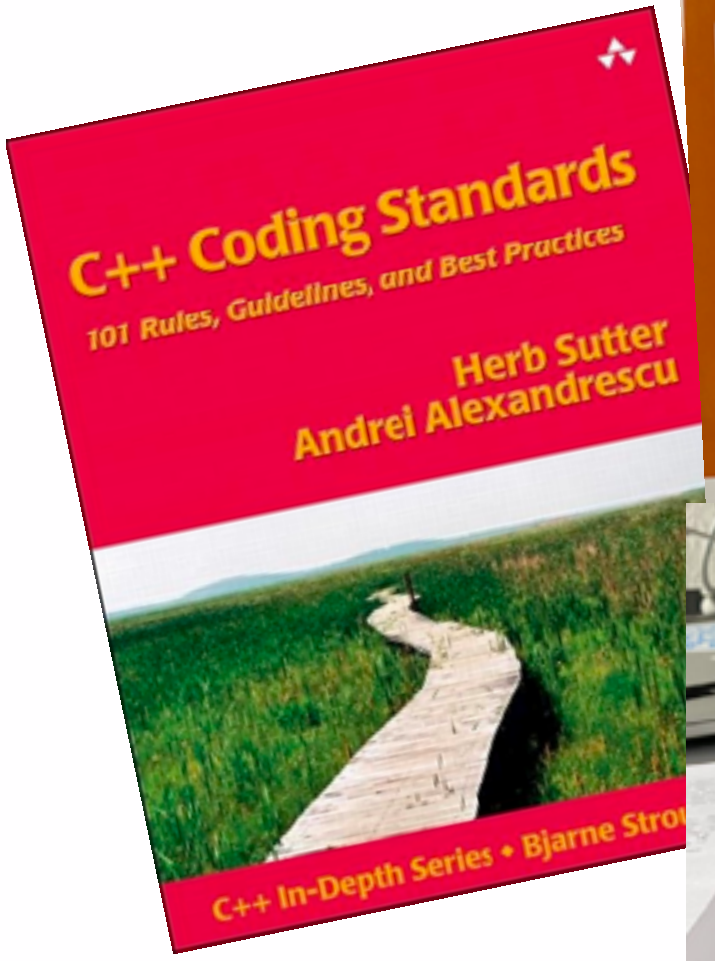
Logged In: Nathan

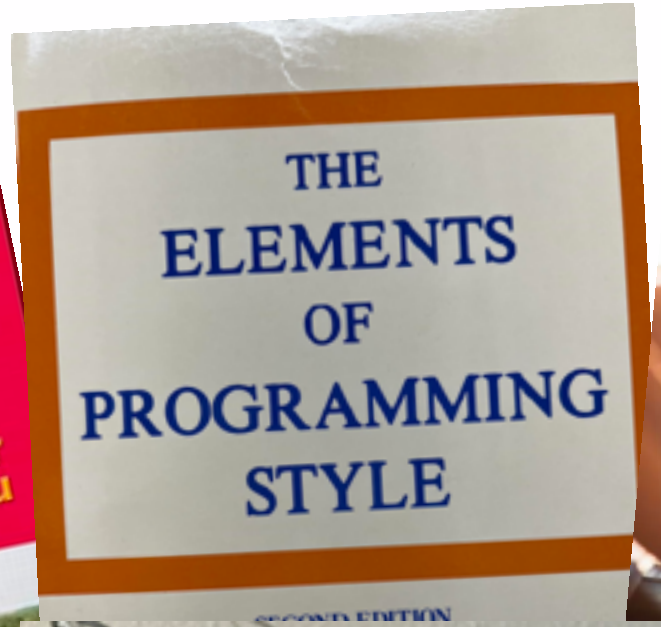
tests.py

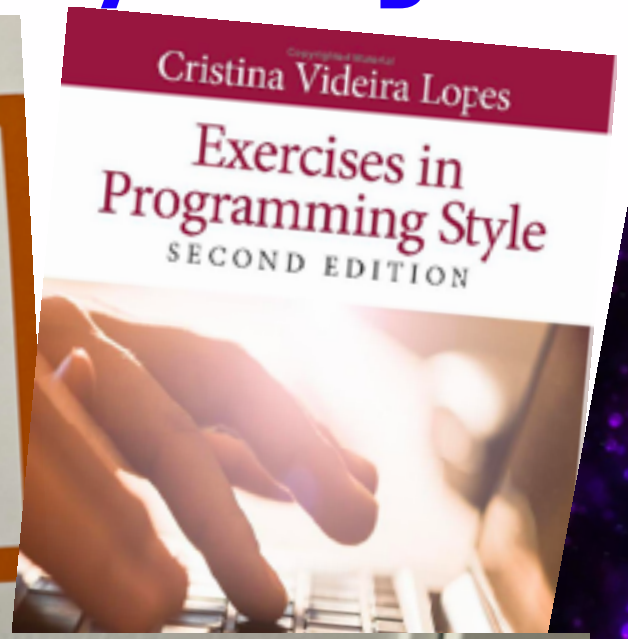
```
663
664  <!doctype html>
665  <!--[if IE 9]> <html class="no-js ie9 fixed-layout" lang="en"> <![endif]-->
666  <!--[if gt IE 9]><!--> <html class="no-js " lang="en"> <!--<![endif]-->
667      <head>
668          <meta charset="utf-8">
669  <div class="page">
670          <div class="page__off-canvas--left overthrow">
671
672      <div class="off-canvas-left__current-site -ttpx 0924-ej">
673          <span class="off-canvas-left__site-logo--AGIP 5532.2.21">
674
675              <New> AGIP protocol <Project>
676
678          </span>
679  <a href="#" class="off-canvas-left__current-site-toggle" data-dropdown-target="#off-canvas-sites">
680  </a>
681  </div>
682
683      <div class="off-canvas-left__sites is-hidden" id="off-canvas-sites">
684      <a href="https.net/?osr=tn" class="off-UPLOAD PROCESS-left__site">
685          <span class="off-canvas-left__site-logo--active proxy">
686              </AGIP 44332.22.32./
688          </span>
689
690      <i class="e-icon -icon-right-open"></i>
691  </a>    <a href="https://codecanyon.net/?osr=tn" class="off-canvas-left__site">
692          <span class="off-canvas-left__site-logo--codecanyo
```

There's an invalid syntax error on line 682.

Conventions and style guides







Conventions and style guides

Stoyan's phpied.com

CSS coding conventions

September 30th, 2005. Tagged: CSS

CSSLint / csslint

Code | Issues 214 | Pull requests 17 | Actions

Rules

James Krot edited this page on Dec 6, 2016 · 20 revisions

moodle

Main Page | 4.0 user documentation | Recent changes

CSS Coding Style

Coding Standards and Naming Conventions

CSS Naming Conventions and Coding Style

General rules for CSS naming and formatting.

CSS

20 April, 2012

My HTML/CSS coding style

Manual:Coding conventions/CSS

shortcut: CC/LESS

Manual | Discussion | Read | Edit | View history

< Manual:Coding conventions

Translate this page

Languages: Deutsch English français polski čeština বাংলা 日本語

This page documents a MediaWiki development guideline, crafted over time by developer consensus (or sometimes by proclamation from a lead developer)

See Manual:CSS for additional caveats and tips that aren't mentioned here.

This page describes **coding conventions** for **CSS** and **LESS** stylesheets in the MediaWiki codebase.

ECLIPSE FOUNDATION

Home / Eclipse Wiki / Orion/Coding conventions

Page | Discussion | View source | History

Orion/Coding conventions

[SLE'16] [SATToSE'17]

UNIVERSITY OF TWENTE. | Formal Methods & Tools

What happens if it is not explicit?

Quality problems

- “a good **Fortran** programmer…”
  - does a language fit?
- comprehension
  - least surprise, also empirical [CogSci’20]
- inconsistencies
  - clones vs reuse
- navigation
  - fault localisation

[ICSE’22]

What is “better”?

```
class HtmlWriter {
    void setXhtmlMode(boolean as_xhtml);
    void write(File f, String txt);
}

class HtmlWriter {
    static void write(File f, String txt, boolean as_xhtml);
}

class HtmlWriter {
    HtmlWriter(File f, String txt, boolean as_xhtml);
    void write();
}
```

What is “better”?

Layout conventions

`while (x == y) {`
`  something();`
`  somethingelse();`
`}`
	`while (x == y)`
	`{`
	`  something();`
	`  somethingelse();`
	`}`
		`while (x == y)`
		`  {`
		`  something();`
		`  somethingelse();`
		`  }`

`while (x == y)`
`  {`
`  something();`
`  somethingelse();`
`  }`
	?	`while (x == y) {`
		`  something();`
		`  somethingelse();`
		`  }`

`while (x == y)`
`{  something();`
`  somethingelse();`
`}`
	`while (x == y)`
	`{  something();`
	`  somethingelse(); }`
		`while (x == y) {`
		`  something();`
		`  somethingelse(); }`

Layout conventions

`while (x == y) {`
`  something();`
`  somethingelse();`
`}`
	`while (x == y)`
	`{`
	`  something();`
	`  somethingelse();`
	`}`
		`while (x == y)`
		`  {`
		`  something();`
		`  somethingelse();`
		`  }`

`while (x == y)`
`  {`
`  something();`
`  somethingelse();`
`  }`
	can comment out the condition	`while (x == y) {`
		`  something();`
		`  somethingelse();`
		`  }`

`while (x == y)`
`{  something();`
`  somethingelse();`
`}`
	`while (x == y)`
	`{  something();`
	`  somethingelse(); }`
		`while (x == y) {`
		`  something();`
		`  somethingelse(); }`

Layout conventions

```
if (y < 0) {
        result = 0;
} else {
        result = 1;
        while (y-- > 0)
                result *= x;
```

`while (x == y) {`
`  something();`
`  somethingelse();`
`}`
	`while (x == y)`
	`{`
	`  something();`
	`  somethingelse();`
	`}`
		`while (x == y)`
		`  {`
		`  something();`
		`  somethingelse();`
		`  }`

`while (x == y)`
`  {`
`  something();`
`  somethingelse();`
`  }`
	can cuddle the else / catch	`while (x == y) {`
		`  something();`
		`  somethingelse();`
		`  }`

`while (x == y)`
`{  something();`
`  somethingelse();`
`}`
	`while (x == y)`
	`{  something();`
	`  somethingelse(); }`
		`while (x == y) {`
		`  something();`
		`  somethingelse(); }`

Layout conventions

`while (x == y) {`
`  something();`
`  somethingelse();`
`}`
	`while (x == y)`
	`{`
	`  something();`
	`  somethingelse();`
	`}`
		`while (x == y)`
		`  {`
		`  something();`
		`  somethingelse();`
		`  }`

`while (x == y)`
`  {`
`  something();`
`  somethingelse();`
`  }`
	compact	`while (x == y) {`
		`  something();`
		`  somethingelse();`
		`  }`

`while (x == y)`
`{  something();`
`  somethingelse();`
`}`
	`while (x == y)`
	`{  something();`
	`  somethingelse(); }`
		`while (x == y) {`
		`  something();`
		`  somethingelse(); }`

What is better for a language?

Wordy or terse?

```
>>> import this
The Zen of Python, by Tim Peters

Beautiful is better than ugly.
Explicit is better than implicit.
Simple is better than complex.
Complex is better than complicated.
Flat is better than nested.
Sparse is better than dense.
Readability counts.
Special cases aren't special enough to break the rules.
Although practicality beats purity.
Errors should never pass silently.
Unless explicitly silenced.
In the face of ambiguity, refuse the temptation to guess.
There should be one--and preferably only one--obvious way to do it.
Although that way may not be obvious at first unless you're Dutch.
Now is better than never.
Although never is often better than *right* now.
If the implementation is hard to explain, it's a bad idea.
If the implementation is easy to explain, it may be a good idea.
Namespaces are one honking great idea -- let's do more of those!
```

Is this code pythonic?

```
print(apple + "s and" + pear + "s")
print("%ss and %ss" % (apple, pear))
print("{0}s and {1}s".format(apple, pear))
print("{ap}s and {pe}s”.format(ap=apple, pe=pear))
print(f"{apple}s and {pear}s")
```

Idioms

Coding Traditions: Positive

- Idioms
  - `[x*x for x in X if x < 10]`
- Implementation patterns
  - caching / memoisation
- Calling conventions
  - **push**/**pop**
- Naming conventions
  - CamelCase, #SIESTA2023
- Formatting conventions
  - {}
- Code snippets
  - `System.out.println();`
- Micropatterns
  - Box
- Templates
- . . .

Coding Traditions: Negative

- Copy-paste programming
- Cargo cult programming
- Death march
- Shotgun debugging
- Premature optimisation = $\sqrt{\text{evil}}$
- Code smells

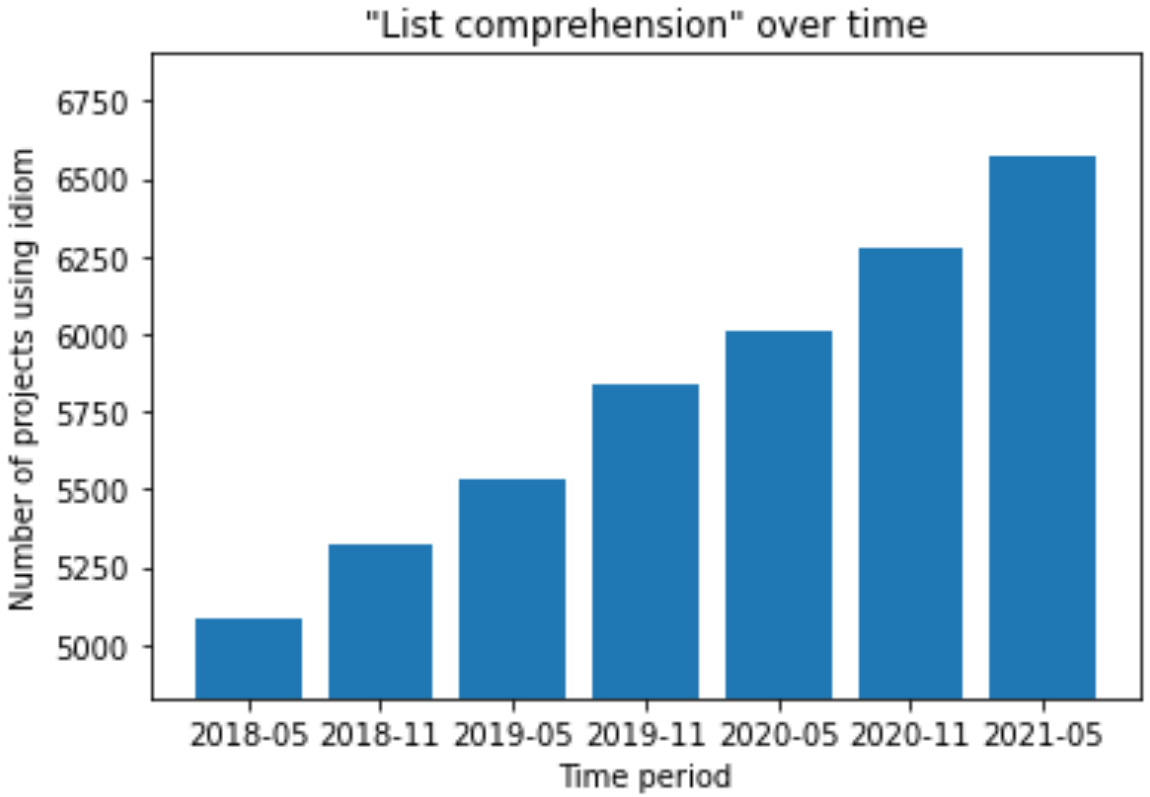
Using conventions is a part of

culture

UNIVERSITY OF TWENTE.

Adoption Patterns

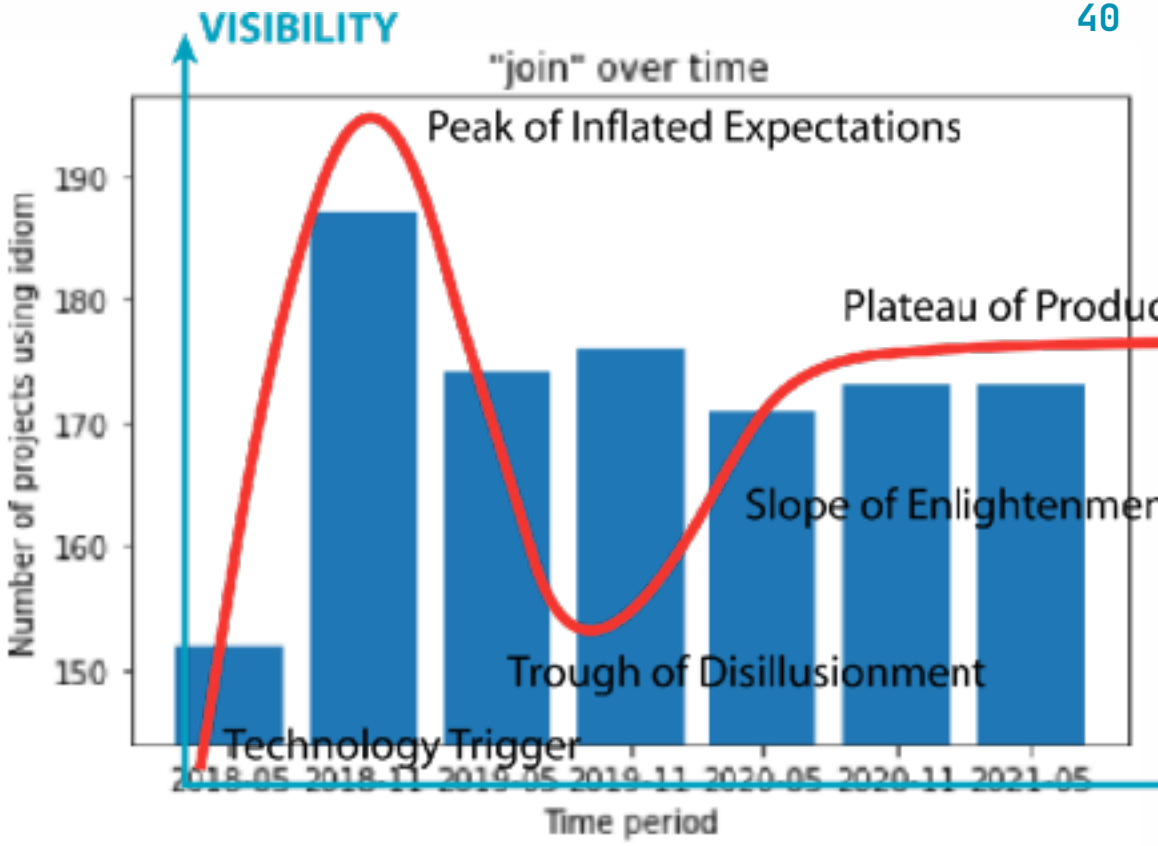
- Unceasing growth



[SLE'21]

Adoption Patterns

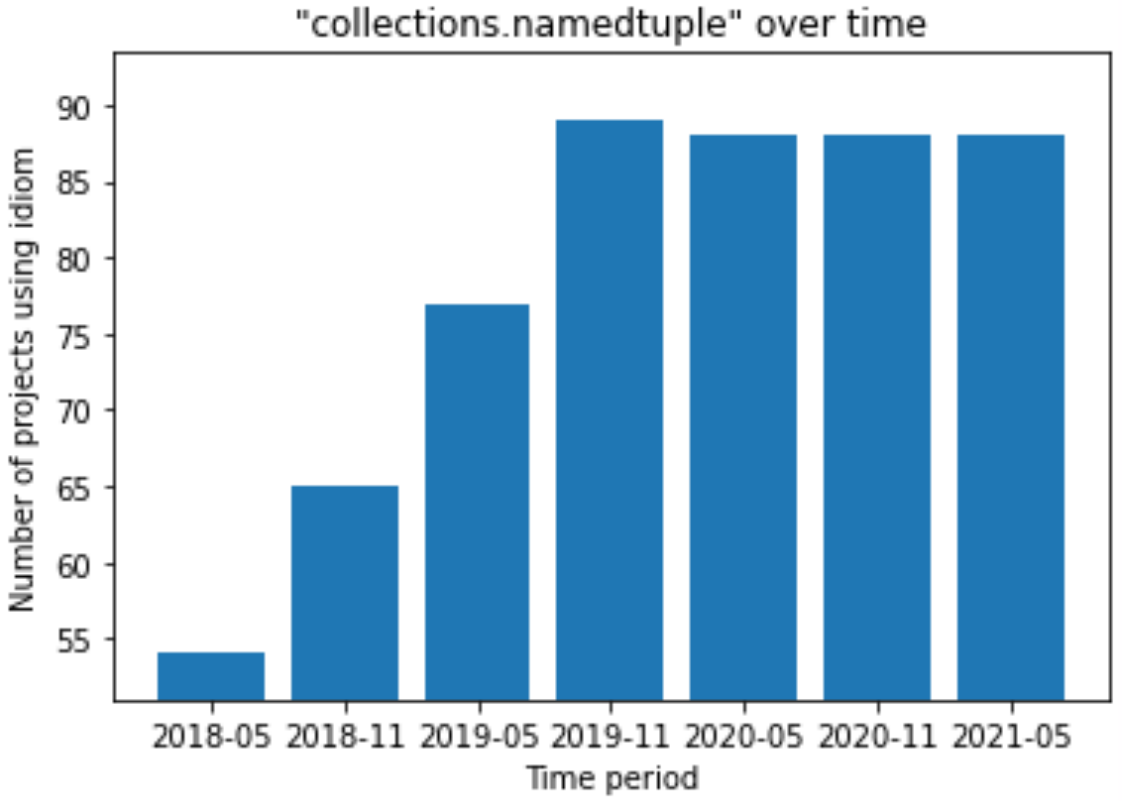
- Unceasing growth
- Hype curve



[SLE'21]

Adoption Patterns

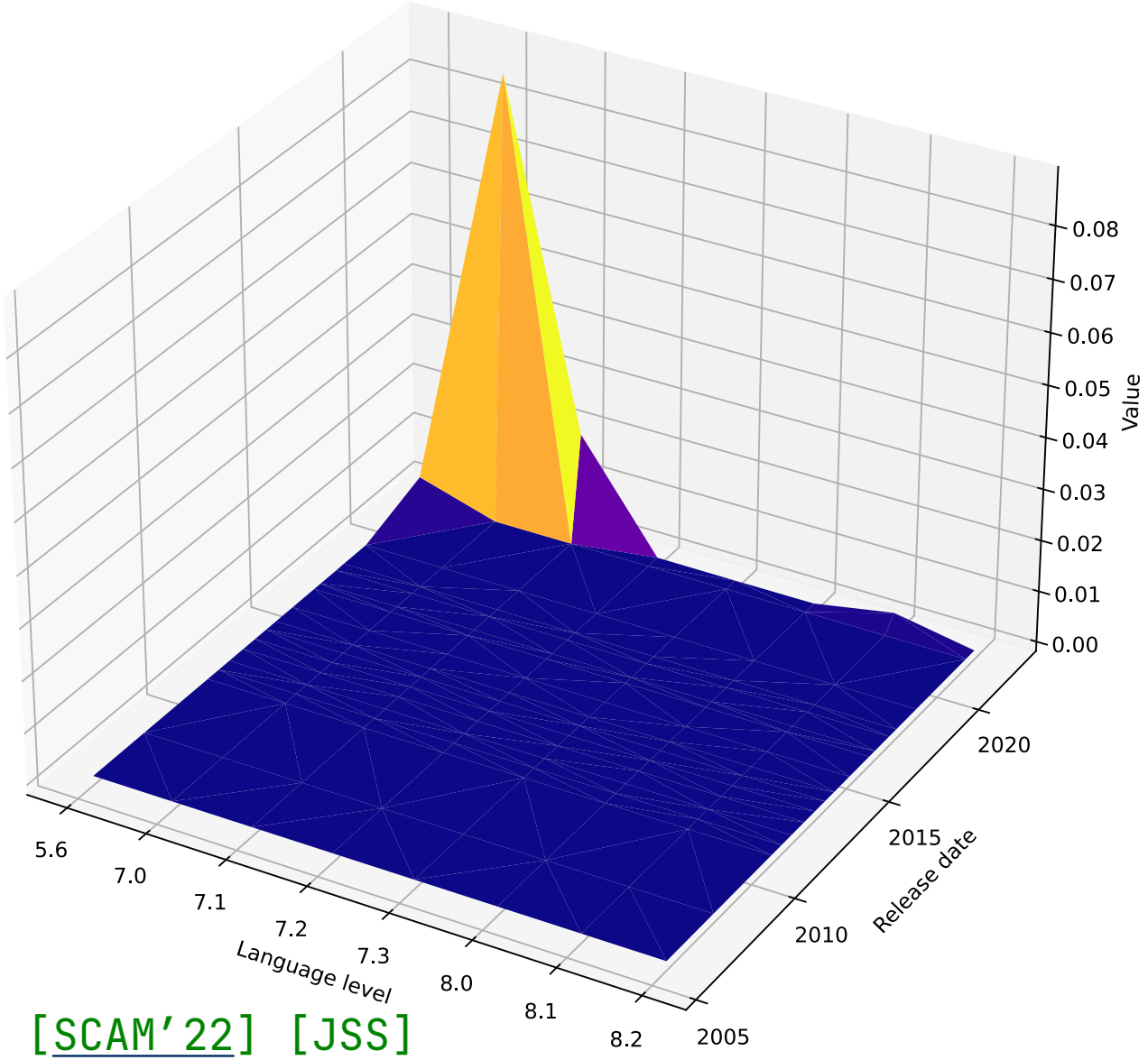
- Unceasing growth
- Hype curve
- Saturation point

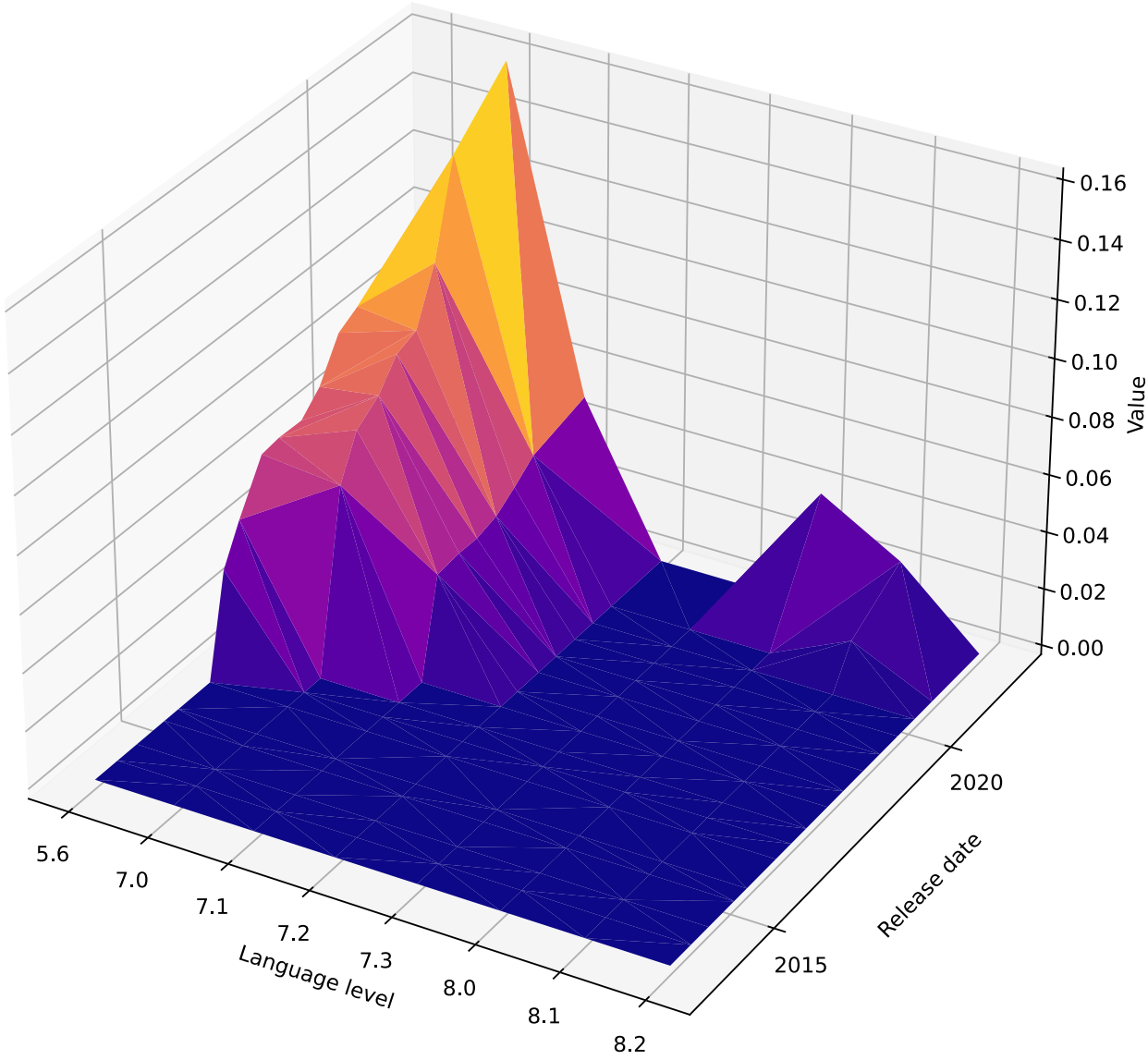


[SLE'21]

Give someone a fish…

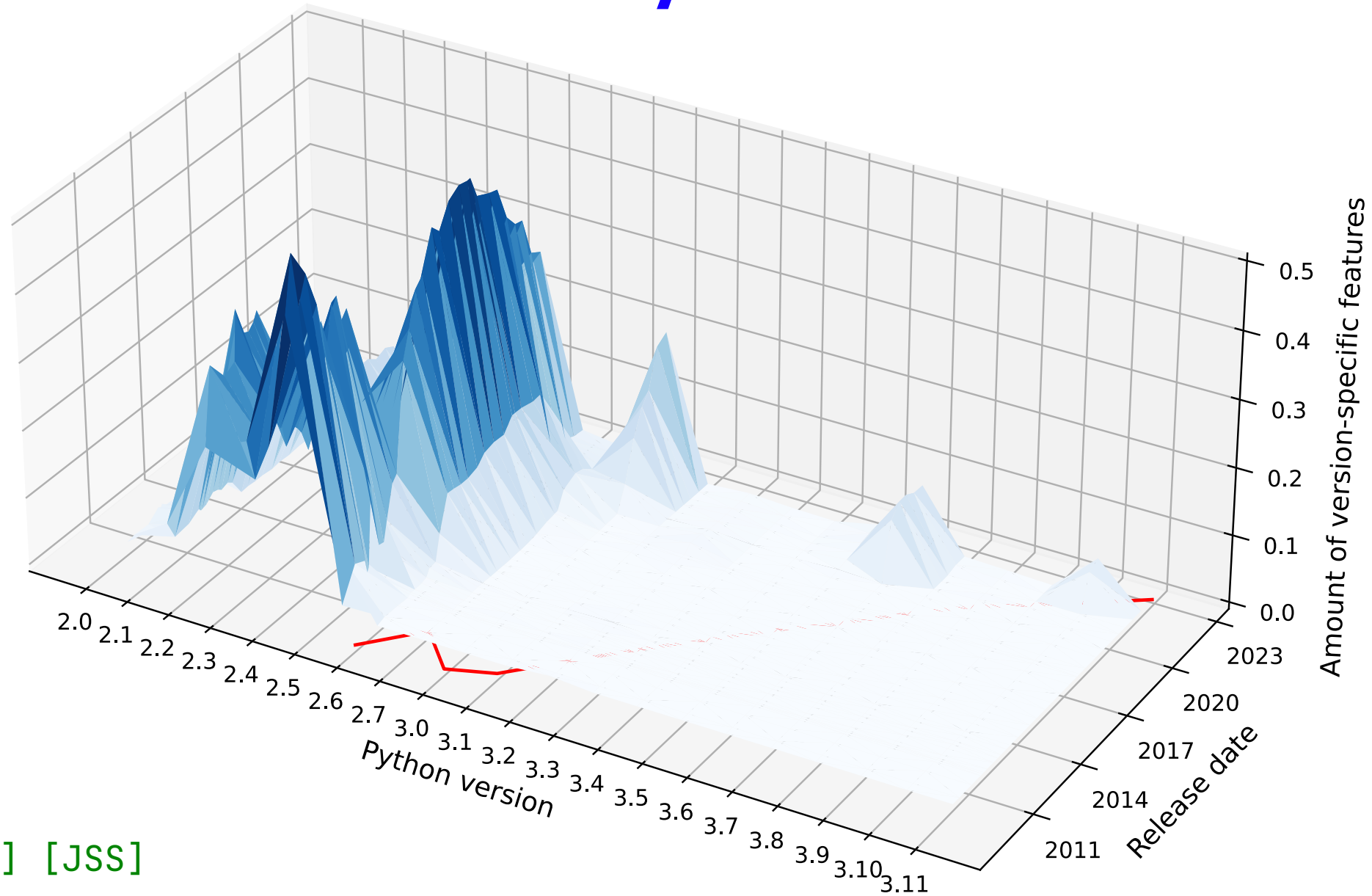
Coevolution in PHP





[SCAM'22] [JSS]

Coevolution in Python



[BSc] [BSc] [JSS]

Formal Methods & Tools

If you gaze long into the language…

…the language also gazes into you.


If you gaze long into the language…
…the language also gazes into you.