

# Evolution of TC39 Proposals



Kai Waløen

University of Bergen

TC39-TG5 Meeting 30 April 2025

Project Supervisor: Mikhail Barash

# Project Aims



- Collect proposal data
  - Snapshot from Feb 2025
  - From TC39/proposals repositories
- Classify proposals
  - Stage
  - Type of change
  - Topics and keywords
  - Present on website
- Analyse evolution of proposals
  - Pattern in stage bumps?
  - Pattern in stage duration?
  - Topic dependent?
  - Time dependent?

```
Stage 4
Classification: Syntactic Change Semantic Change
Human Validated: KW
Title: RegExp v flag with set notation + properties of strings
Authors: Markus Scherer, Mathias Bynens
Champions: Mathias Bynens
Last Presented: May 2023
Stage Upgrades:
Stage 1: 2021-01-28
Stage 2: 2021-05-27
Stage 2.7: NA
Stage 3: 2022-03-29
Stage 4: 2023-05-16
Last Commit: 2023-09-22
Topics: #regex #others #collections
Keywords: #regex #flag #string #set
GitHub Link: https://github.com/tc39/proposal-regexp-v-flag
GitHub Note Link: https://github.com/tc39/notes/blob/HEAD/meetings/2023-05/may-16.md#regexp-v-flag-for-stage-4
```

<https://js-proposals.vercel.app/>

# How?



- Data:
  - Retrieve data via Github API – TC39/Proposals
  - Parse the data, create .md files and saved in Obsidian
  - GPT assistance – Classifications, Stage bumps from commit messages, keywords
  - Manually verified and curated
  - Data analysis done in R and Rstudio
- Website:
  - Quartz: Open source static page generator with Obsidian compatibility
  - Demonstration

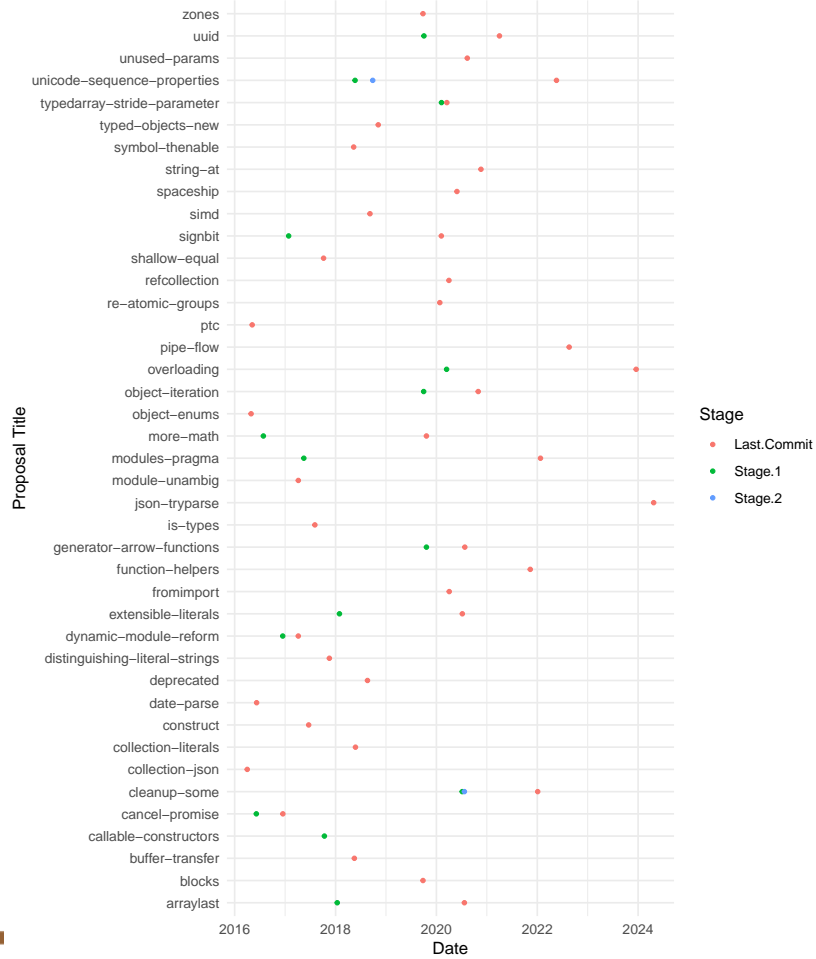
Observations



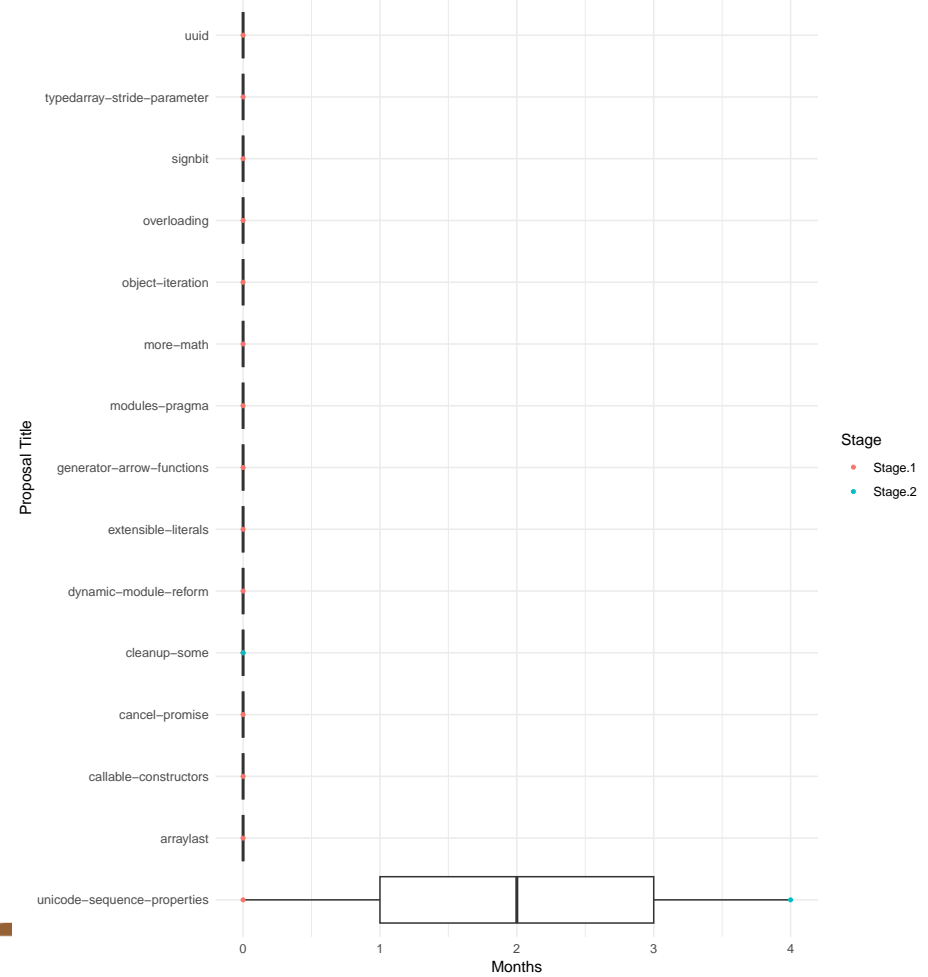
# Inactive



Inactive Proposal Timeline by Title

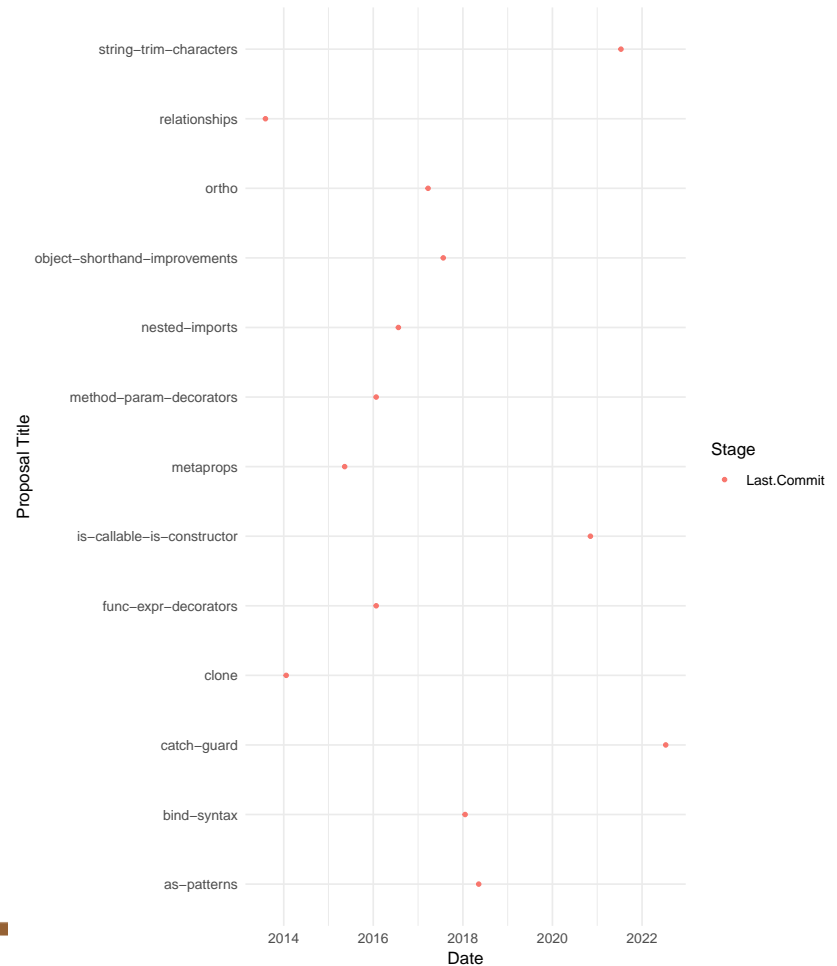


Inactive Proposals Date Spread per Proposal



# Stage 0

Stage 0 Proposal Timeline by Title



# Stage 1

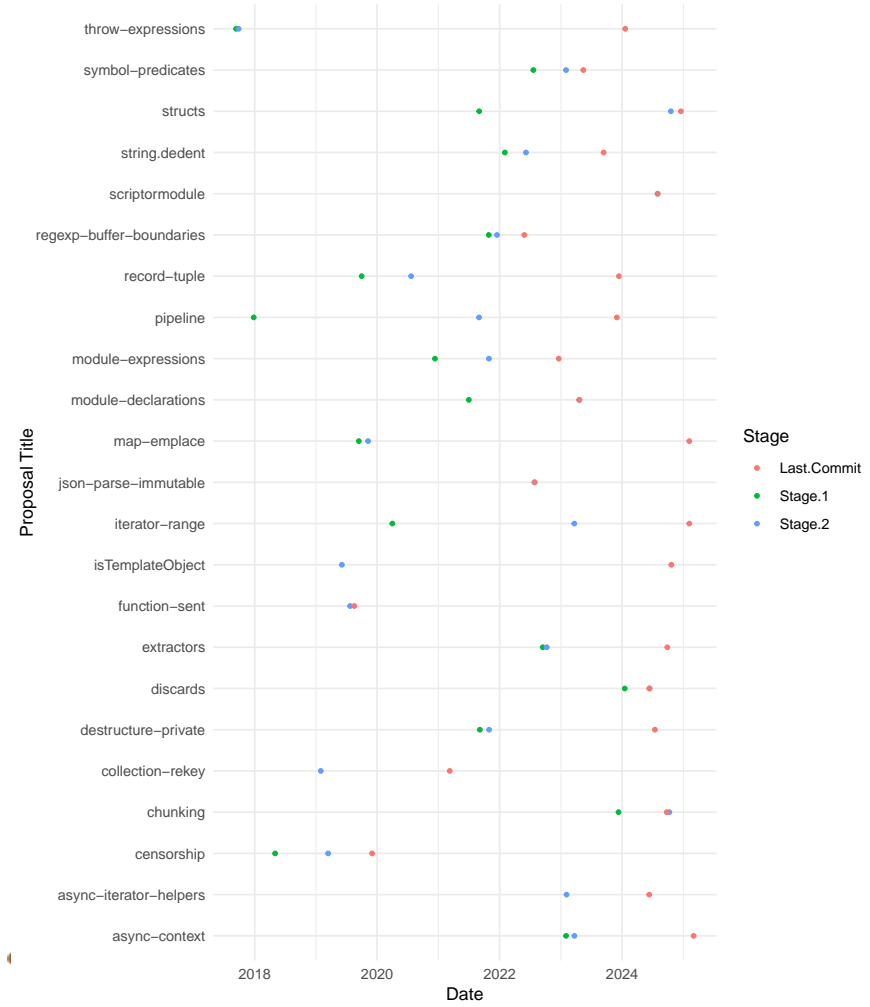
Stage 1 Proposal Timeline by Title



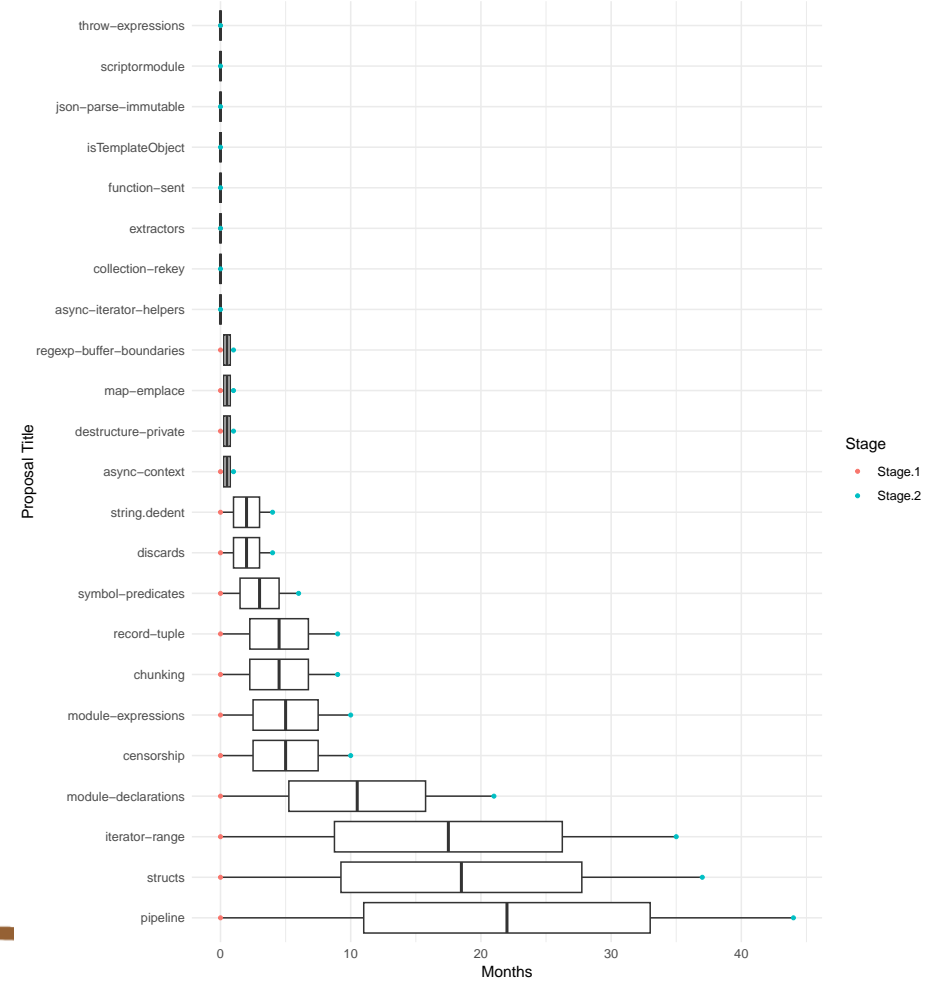
# Stage 2



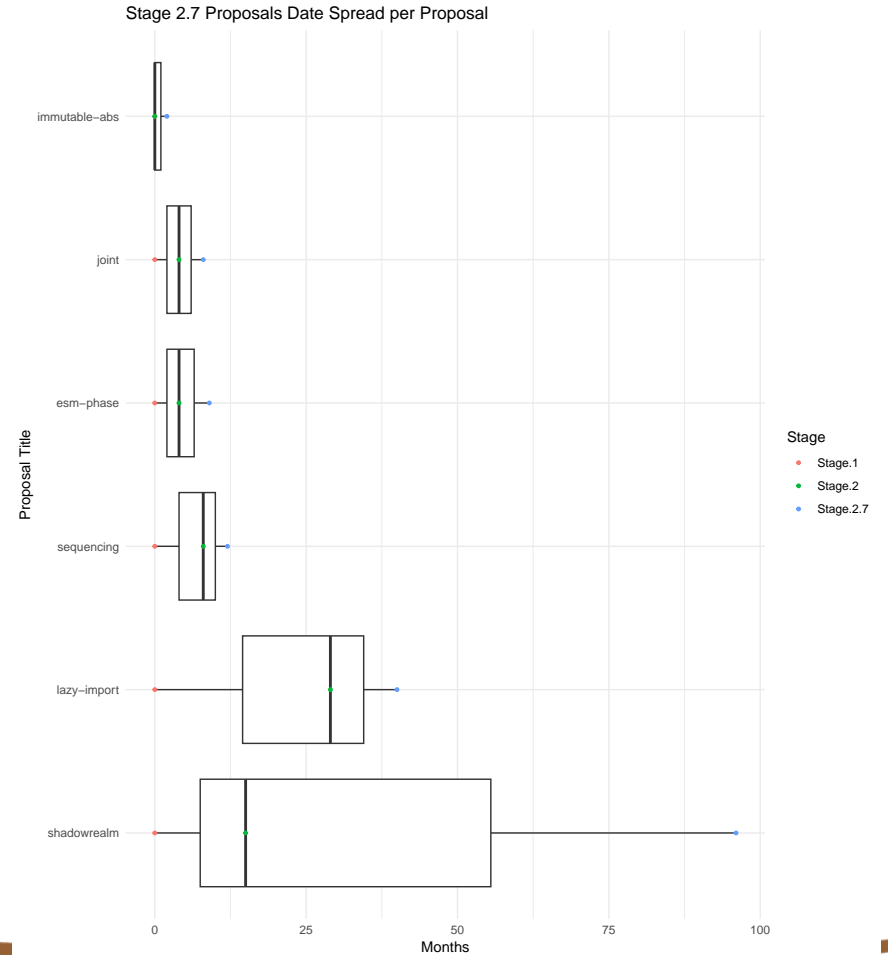
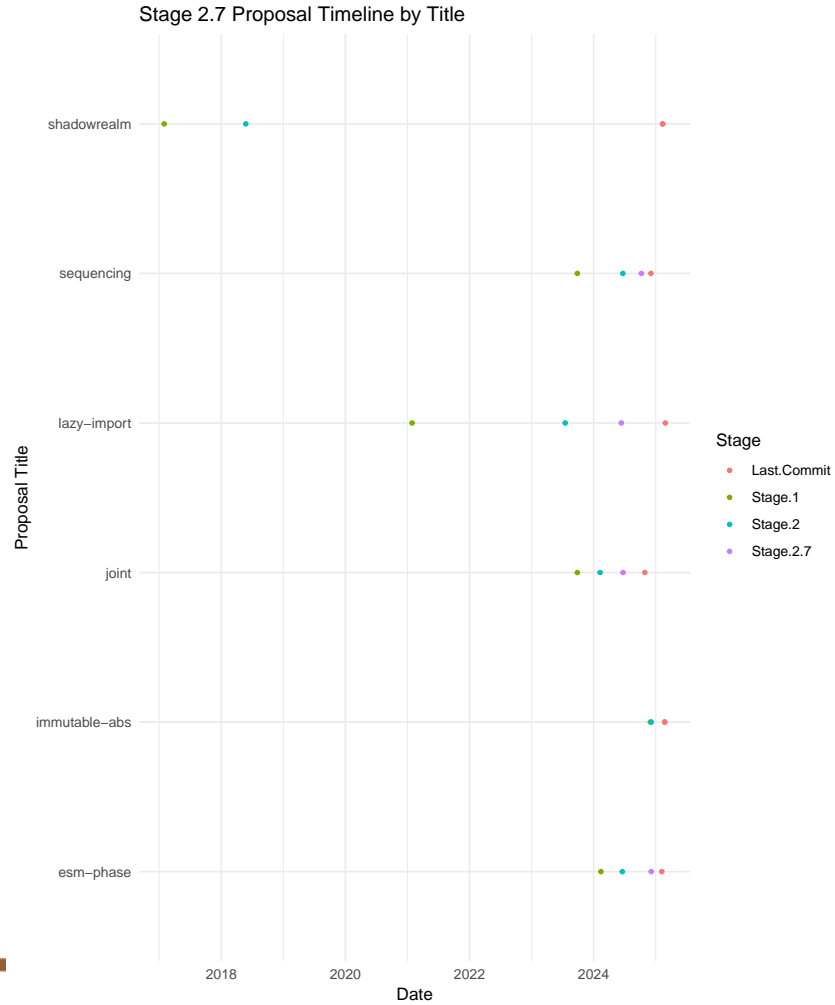
Stage 2 Proposal Timeline by Title



Stage 2 Proposals Date Spread per Proposal



# Stage 2.7

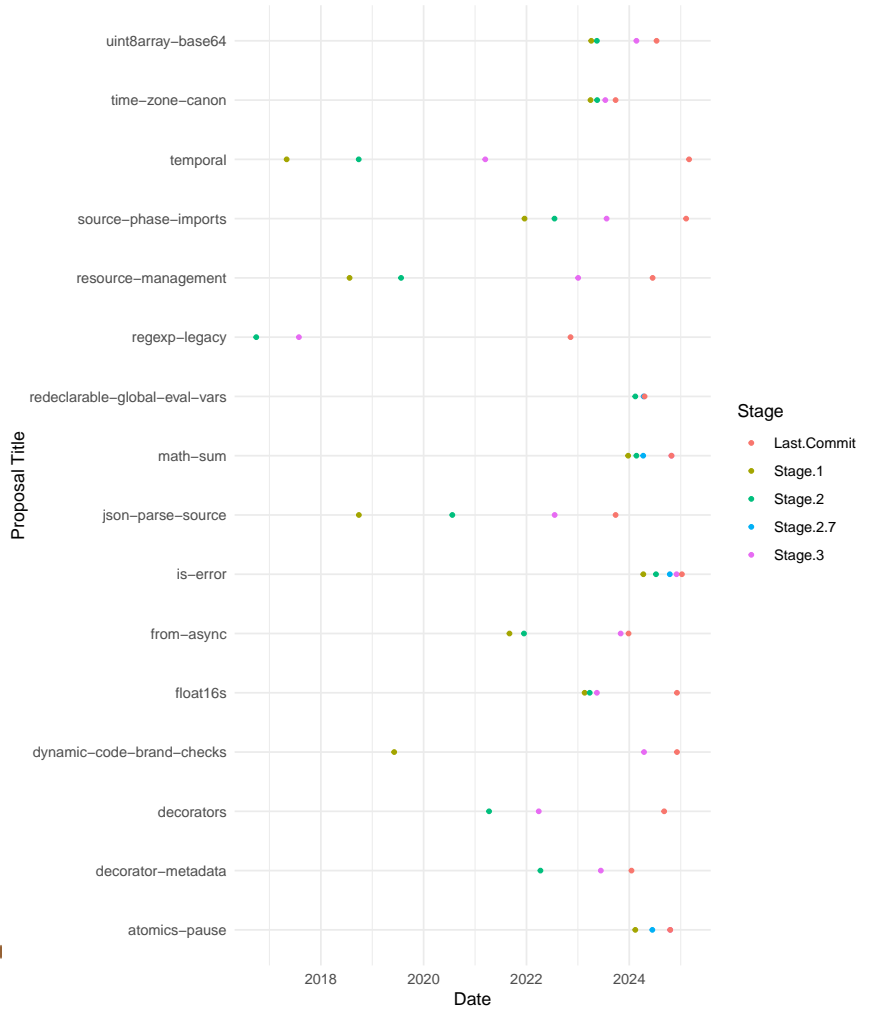




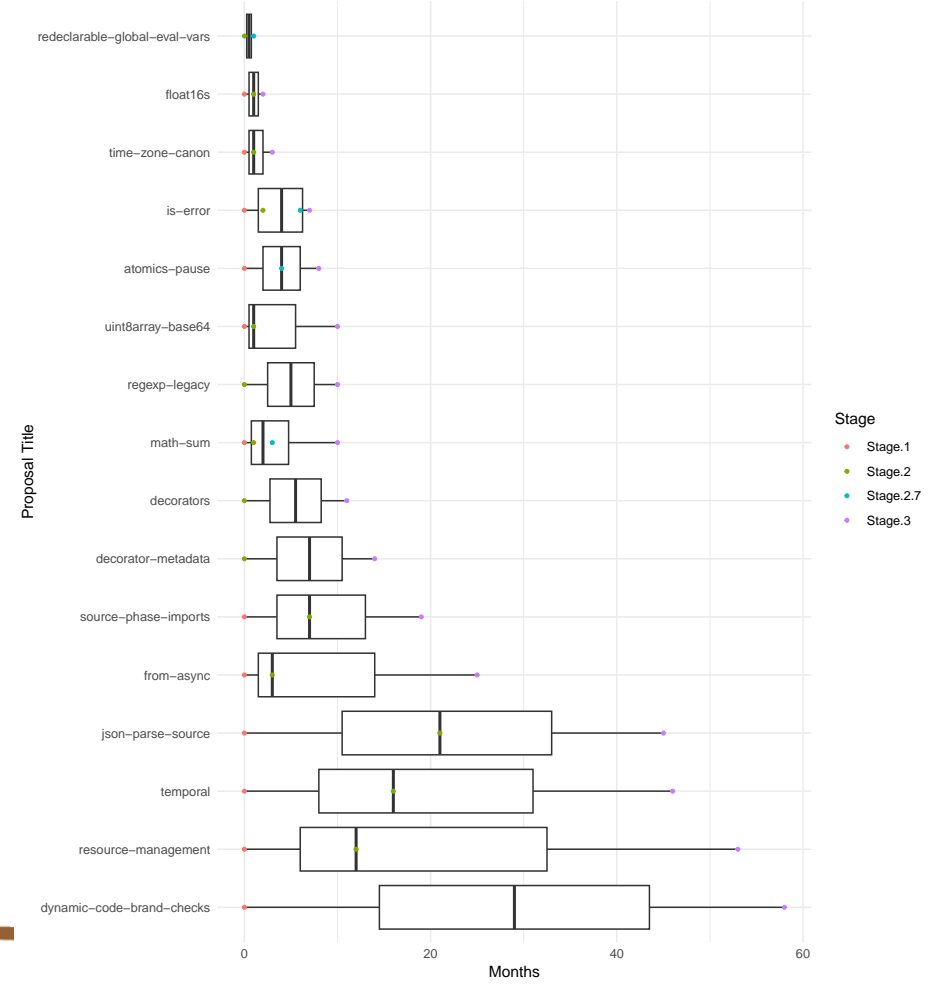
# Stage 3



Stage 3 Proposal Timeline by Title



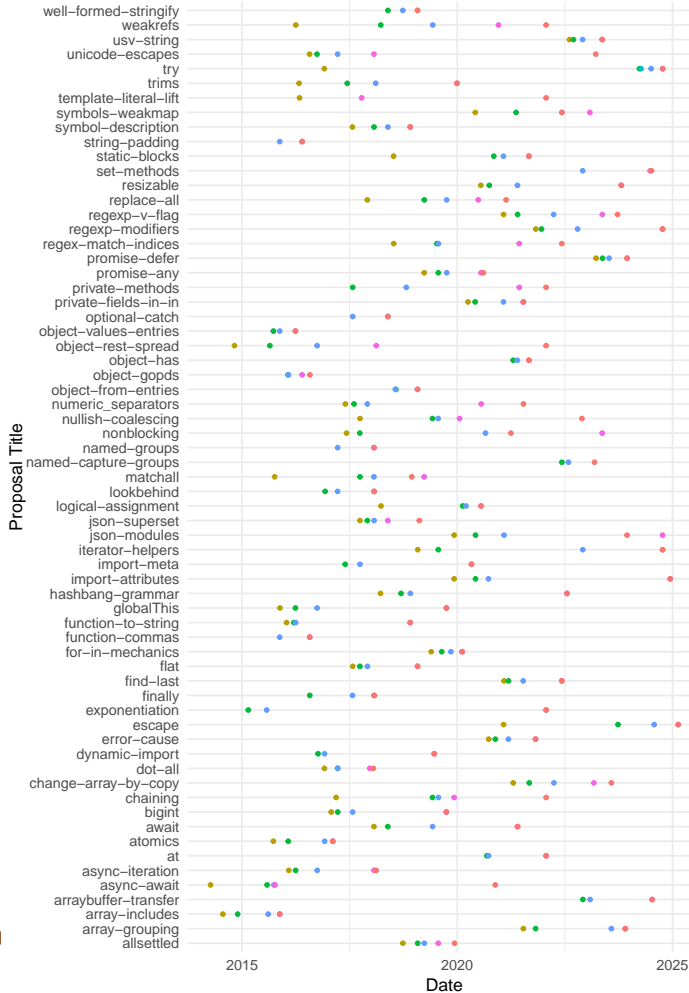
Stage 3 Proposals Date Spread per Proposal



# Stage 4



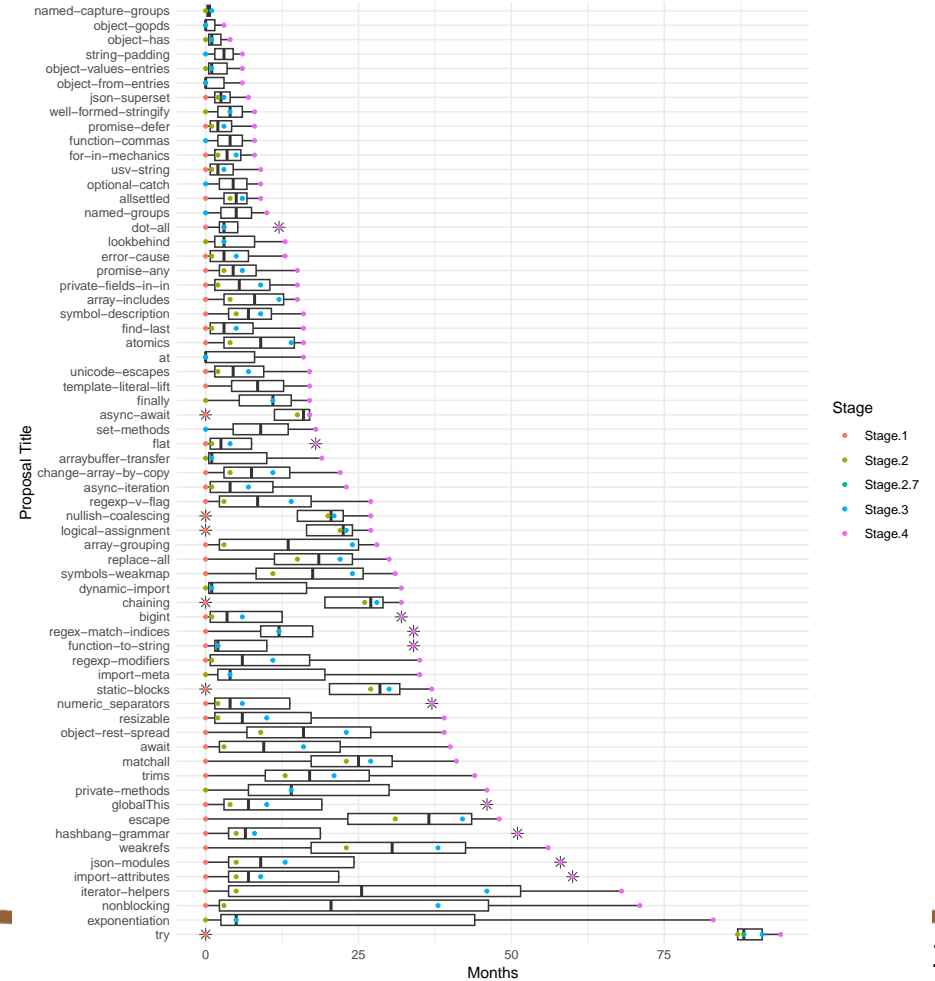
Stage 4 Proposal Timeline by Title



## Stage

- Last.Commit
- Stage.1
- Stage.2
- Stage.2.7
- Stage.3
- Stage.4

Stage 4 Proposals Date Spread per Proposal



## Stage

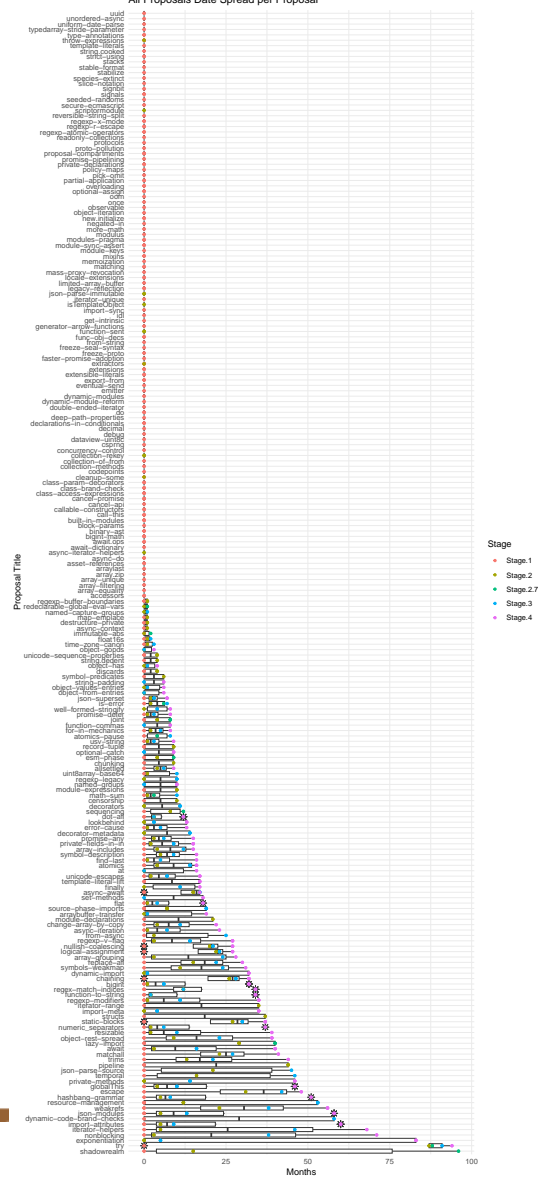
- Stage.1
- Stage.2
- Stage.2.7
- Stage.3
- Stage.4



# All together

What data can be extracted?

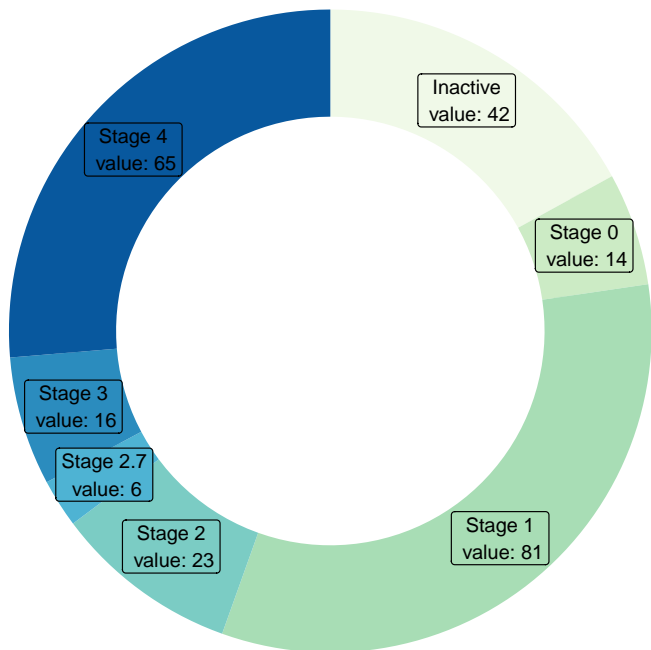
- Classifications
- Stage distribution
- Average duration per stage



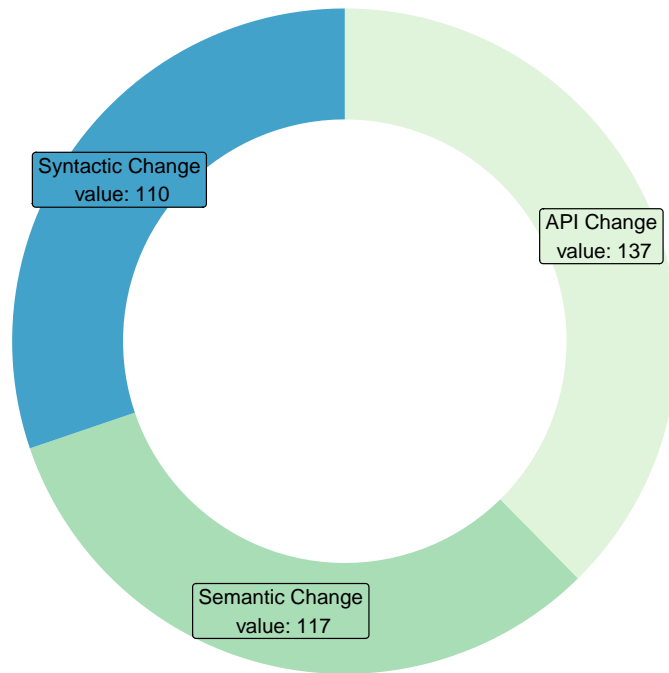
# For starters



- Total number of proposals: 257



- Per Classification:



Note: Proposals can overlap classifications

## Lets look at Stage 4

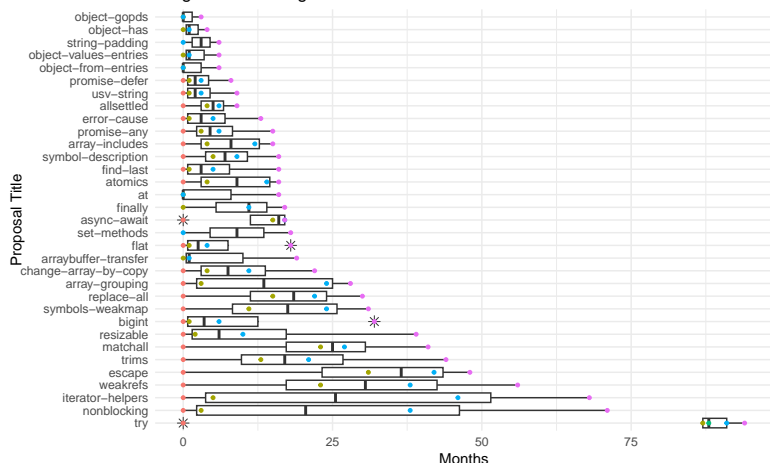


- Most complete data set
- Data gets skewed by the earlier stages

# Stage 4: Average Duration per Change



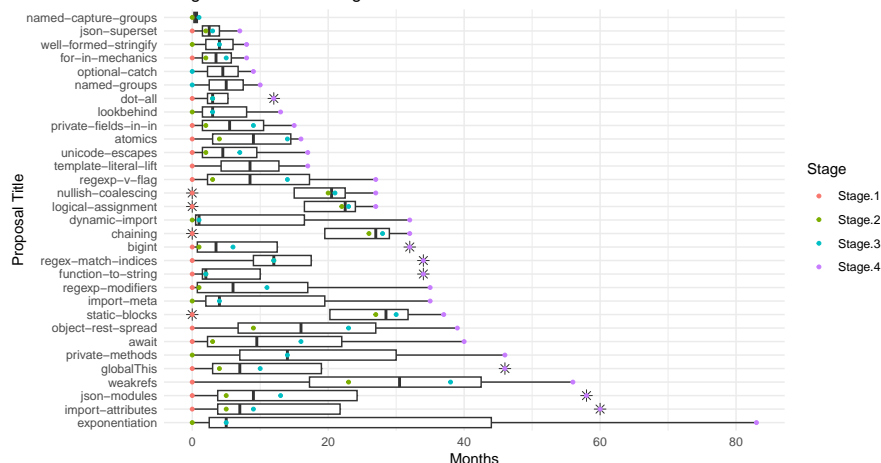
### Stage 4 API Changes



Stage

- Stage.1
- Stage.2
- Stage.2.7
- Stage.3
- Stage.4

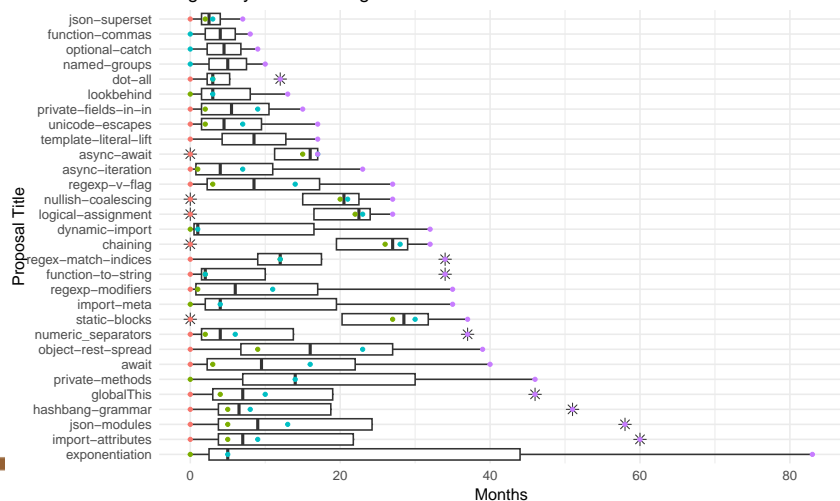
### Stage 4 Semantic Changes



Stage

- Stage.1
- Stage.2
- Stage.3
- Stage.4

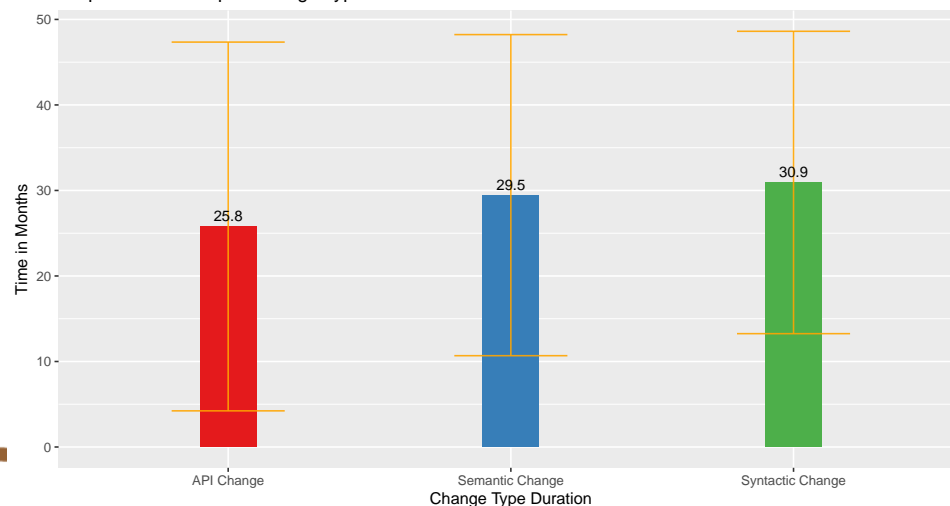
### Stage 4 Syntactic Changes



Stage

- Stage.1
- Stage.2
- Stage.3
- Stage.4

### Proposal Duration per Change Type



# Lets look at more granular classifications



- API only
- Semantic only
- Syntactic only
- API and Semantic
- API and Syntactic
- Semantic and Syntactic
- API and Semantic and Syntactic

# Proposal Categories: Explainer

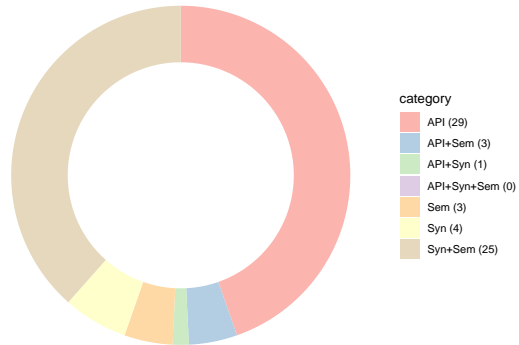
	Category	Description
Primary categories	<b><i>API Proposal</i></b>	<b>modifications to the standard library</b> (built-ins and their observable behavior)
	<b><i>Syntactic Proposal</i></b>	<b>ergonomic grammar enhancements</b> (syntactic sugar to improve the ergonomics of the language)
	<b><i>Semantic Proposal</i></b>	<b>modifications to the interpretation, evaluation, or observable behaviour of existing or new language constructs,</b> independent of concrete syntax or standard library APIs
Composite categories	<i>Syntactic + Semantic</i>	modifications to the interpretation, evaluation, or observable behaviour of existing or new language constructs, <i>together with modifications to the concrete syntax</i>
	<i>Syntactic + API</i>	modifications to the standard library with (ergonomic) grammar enhancements
	<i>Semantic + API</i>	modifications to the interpretation, evaluation, or observable behaviour of existing or new language constructs, together with modifications to the standard library
	<i>Syntactic + Semantic + API</i>	modifications to the interpretation, evaluation, or observable behaviour of existing or new language constructs, together with modifications to the concrete syntax and to the built-ins and/or their observable behaviour



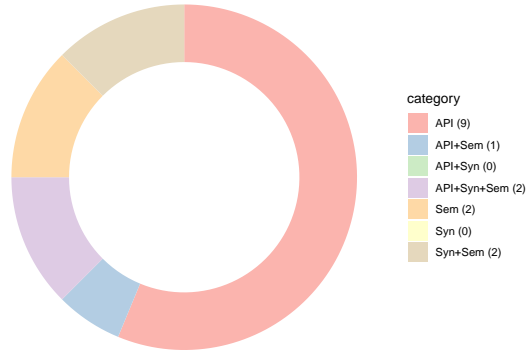
# Specific Classifications



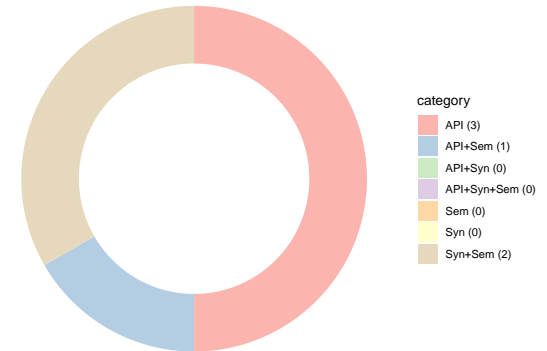
Specific Classification Distribution at Stage 4



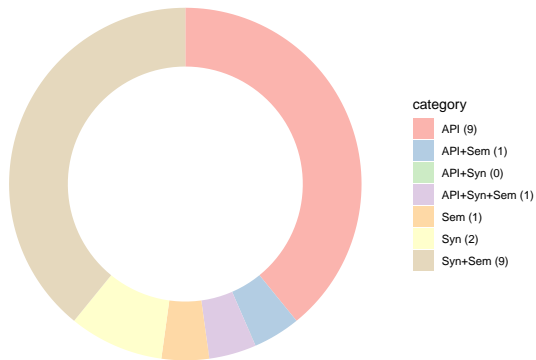
Specific Classification Distribution at Stage 3



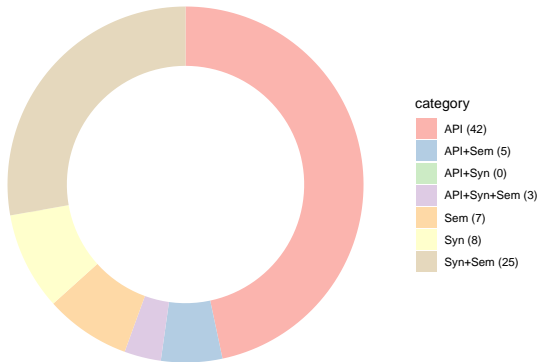
Specific Classification Distribution at Stage 2.7



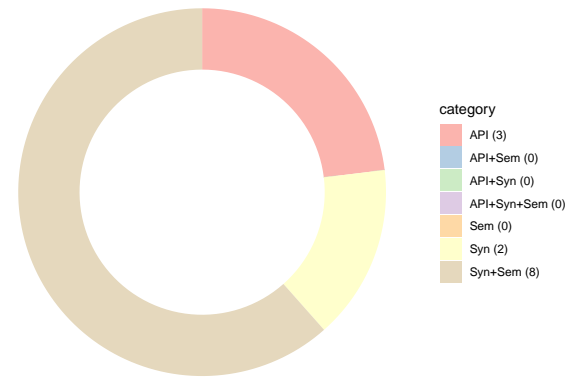
Specific Classification Distribution at Stage 2



Specific Classification Distribution at Stage 1



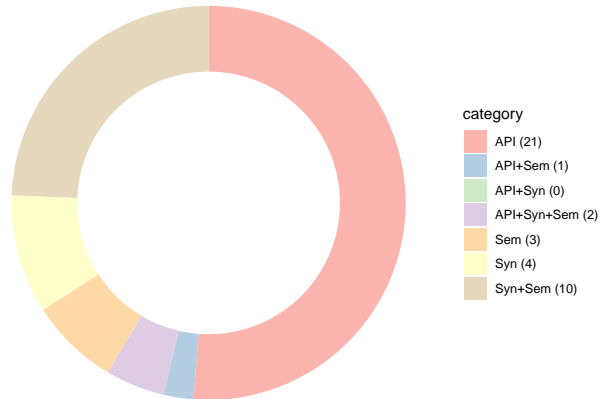
Specific Classification Distribution at Stage 0



# Continuing with Inactive



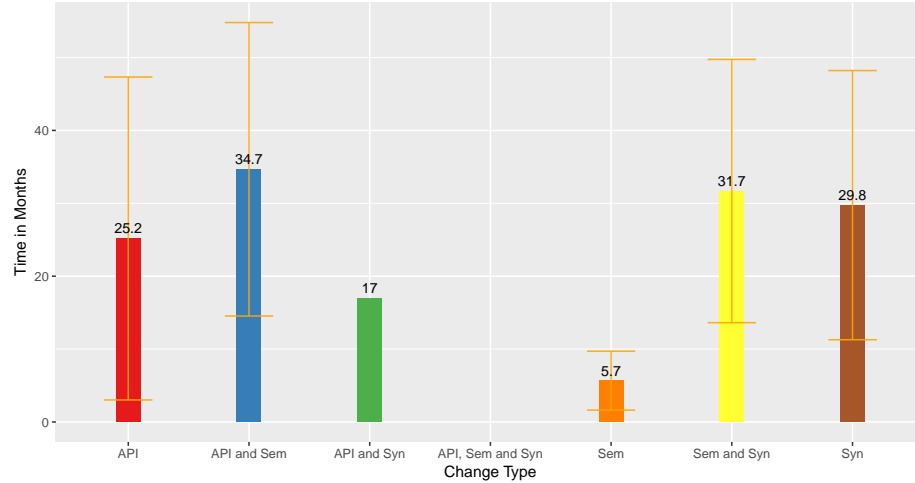
Specific Classification Distribution at Inactive



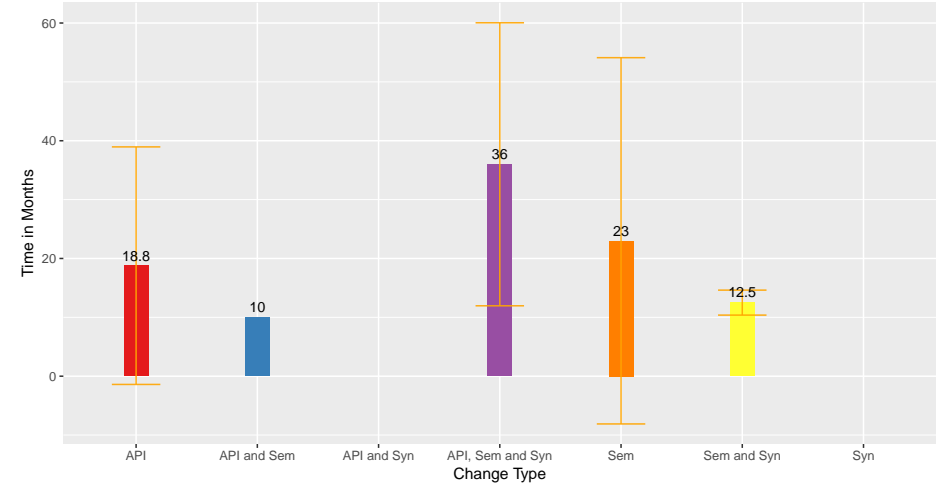
# Time From Stage 1



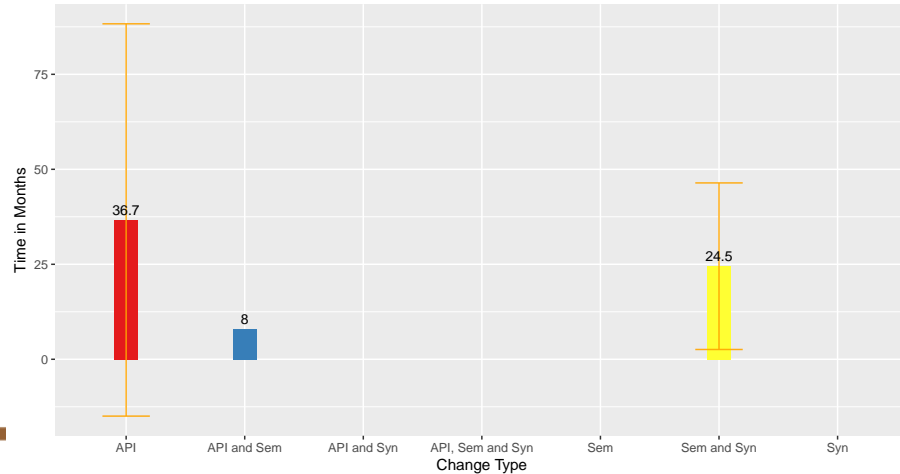
Stage 4: Time from Stage 1 to Stage 4



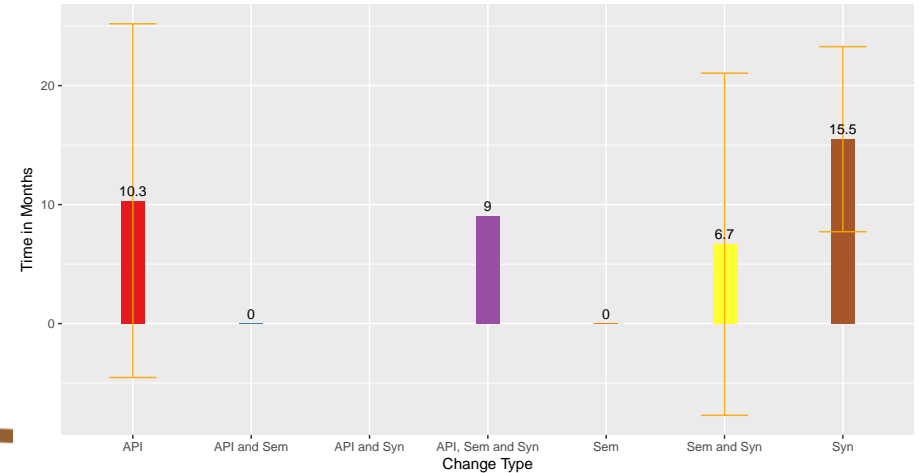
Stage 3: Time from Stage 1 to Stage 3



Stage 2.7: Time from Stage 1 to Stage 2.7



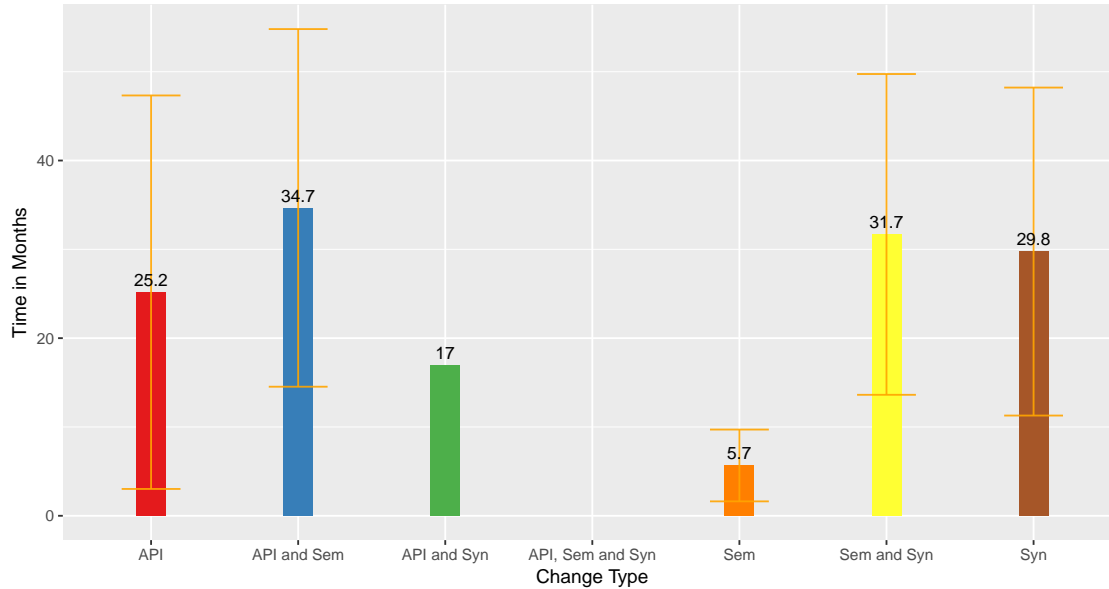
Stage 2: Time from Stage 1 to Stage 2



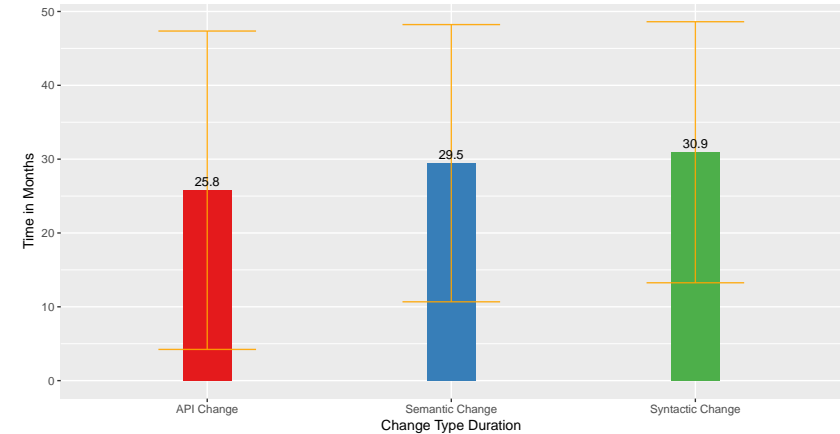
# Comparison granular vs overlapping classification



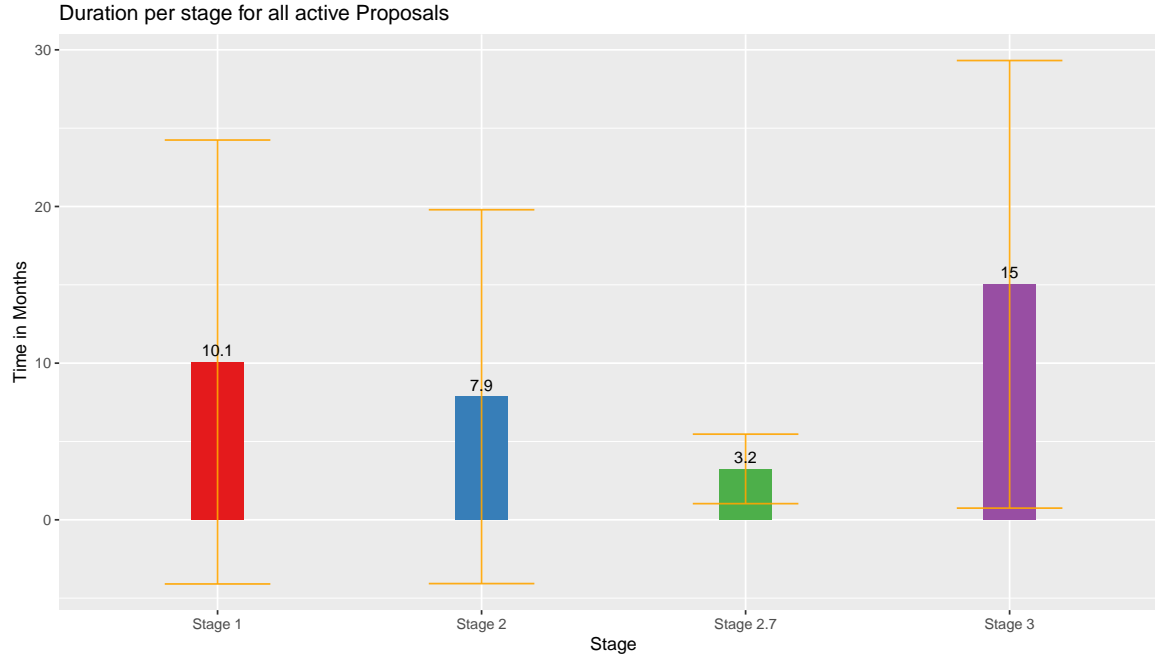
Stage 4: Time from Stage 1 to Stage 4



Proposal Duration per Change Type



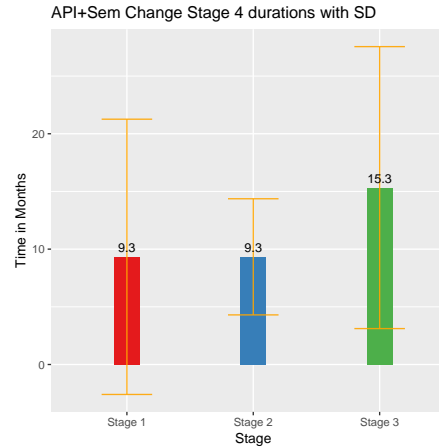
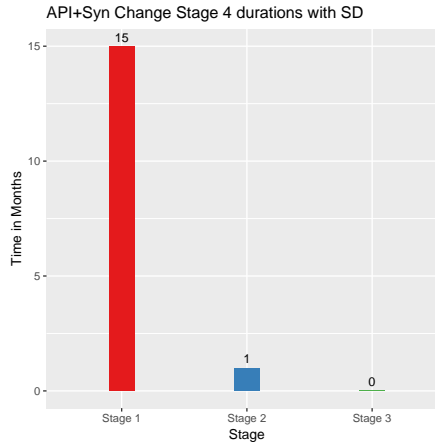
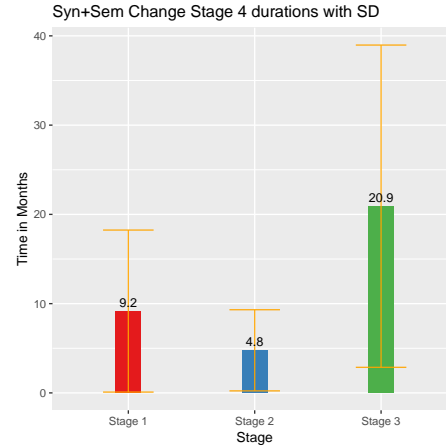
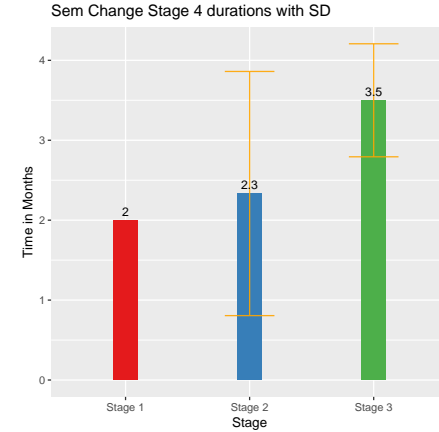
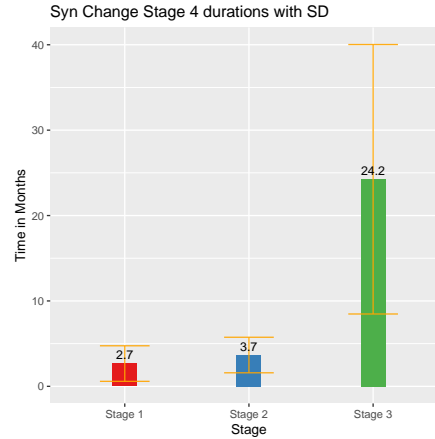
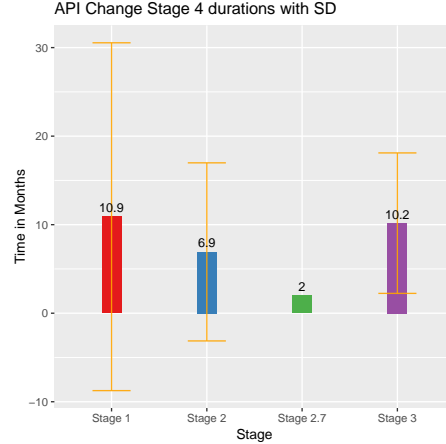
# Duration per Stage



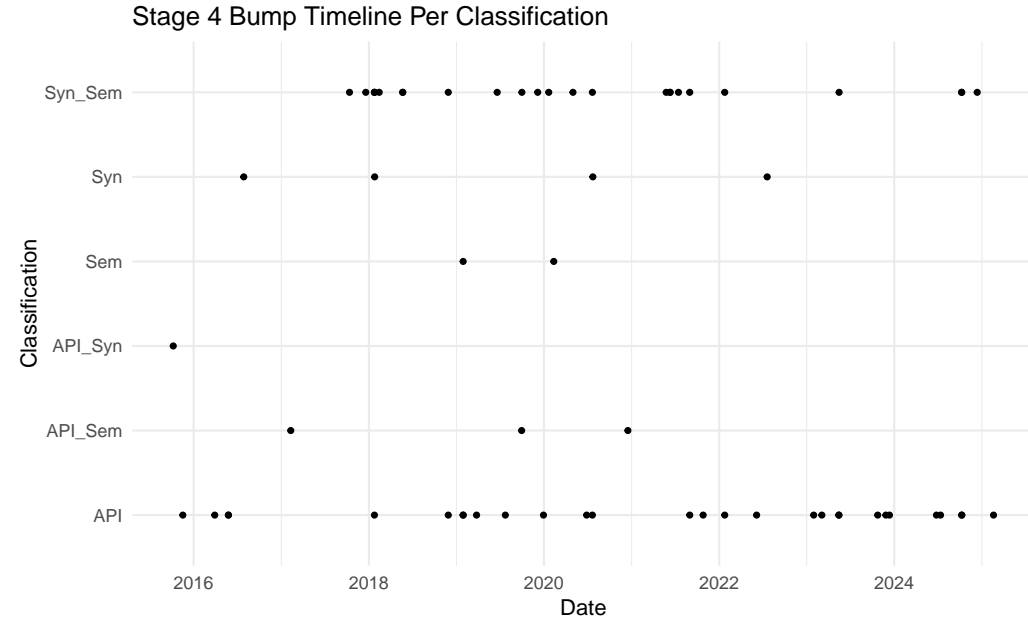
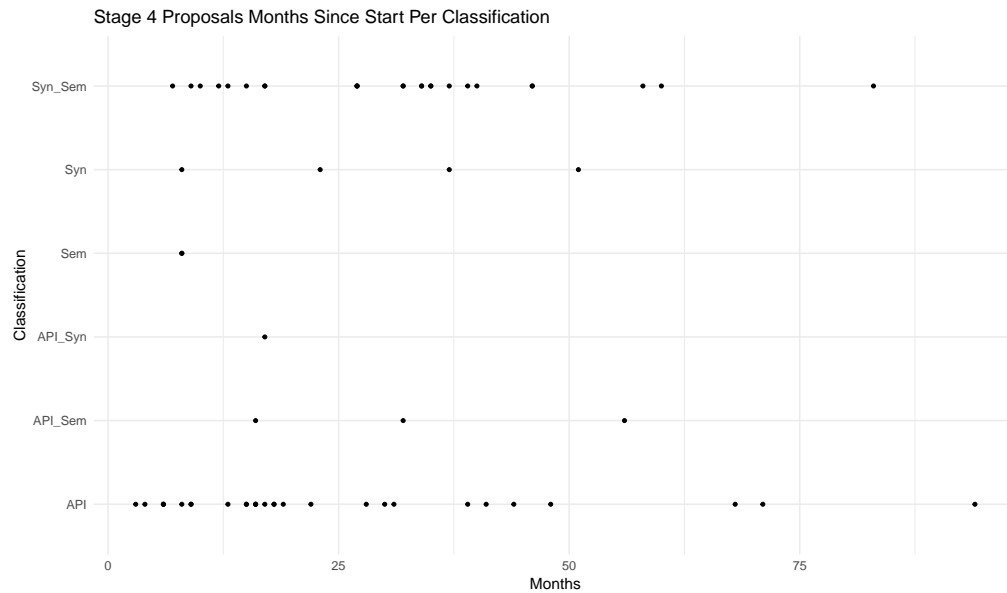
## Observations:

- Length Stage 3 → Stage 1 → Stage 2 → Stage 2.7
- Large SD
- Stage 2.7 is the smallest group

# Durations per stage for Stage 4 per classification



# Stage 4 Proposals per Classifications



# 326 Keywords



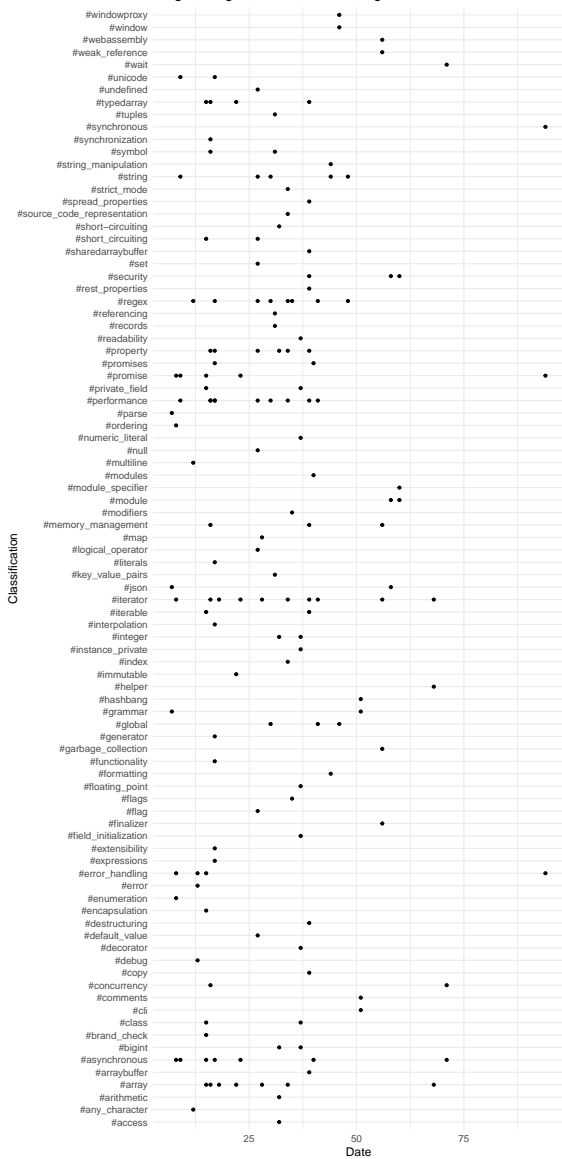
Rank	Keywords	n
1	#performance	32
2	#iterator	26
3	#asynchronous	22
4	#promise	22
5	#module	21
6	#regex	20
7	#array	19
8	#property	19
9	#class	18
10	#security	18
11	#string	17
12	#error_handling	15
13	#memory_management	15
14	#typedarray	12
15	#concurrency	10

Rank	Keywords	n
16	#arithmetic	9
17	#destructuring	8
18	#map	8
19	#numeric	8
20	#arraybuffer	7
21	#decorator	7
22	#json	7
23	#math	7
24	#unicode	7
25	#bigint	6
26	#generator	6
27	#global	6
28	#iterable	6
29	#key_value_pairs	6
30	#parse	6

Rank	Keywords	n
31	#realm	6
32	#symbol	6
33	#date_time	5
34	#encapsulation	5
35	#grammar	5
36	#metadata	5
37	#operator	5
38	#pattern_matching	5
39	#readability	5
40	#resource_management	5
41	#set	5
42	#string_manipulation	5
43	#synchronous	5
44	#wait	5
45	#accessor	4



Stage 4: Tags vs Months Since Stage 1



# Keywords Continued

- Too many keywords
- Reduce to 20

Stage 4: Tags vs Adoption Date



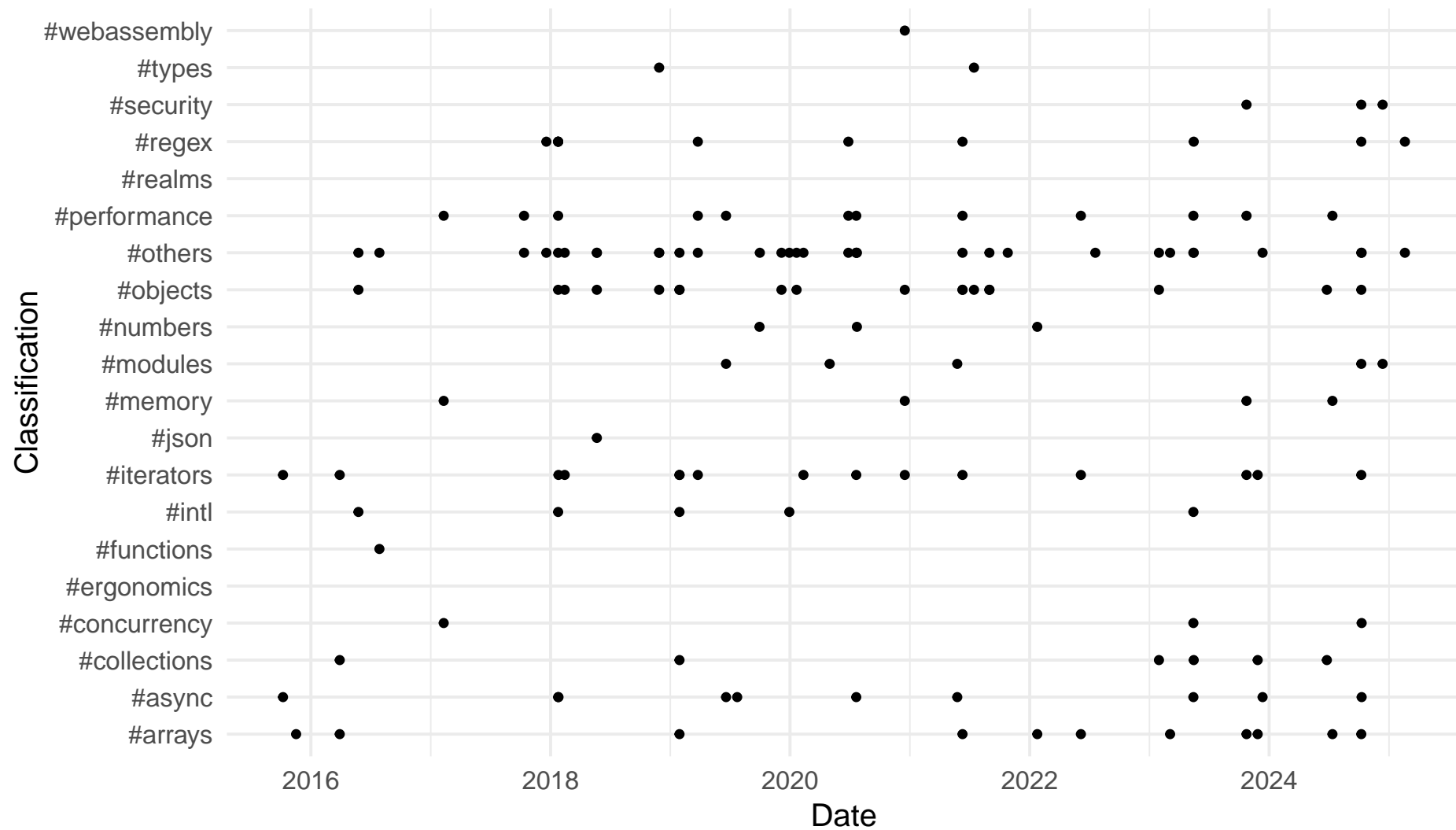
# Topics



- Topics are broader than keywords
- Keywords are more individual
- Can be refined but this is a starting point

Rank	Topics	Count
1	#others	281
2	#objects	131
3	#async	51
4	#arrays	47
5	#iterators	45
6	#modules	37
7	#numbers	36
8	#performance	32
9	#concurrency	31
10	#collections	25
11	#regex	25
12	#security	23
13	#memory	22
14	#intl	21
15	#functions	12
16	#types	11
17	#realms	9
18	#ergonomics	8
19	#json	6
20	#webassembly	2

## Stage 4: Topics vs Adoption Date



# Stage 1: Topics vs Start Date

