```
[10, 1, 3].sort((a, b) => a - b);
```

```
→ [10, 1, 3].sort( );
⇠ ▶(3) [1, 10, 3]
```

```
→ [10, 1, 3].sort((a, b) => a - b);
⇠ ▶(3) [1, 3, 10]
```

Package developers often lack feedback on their interfaces.

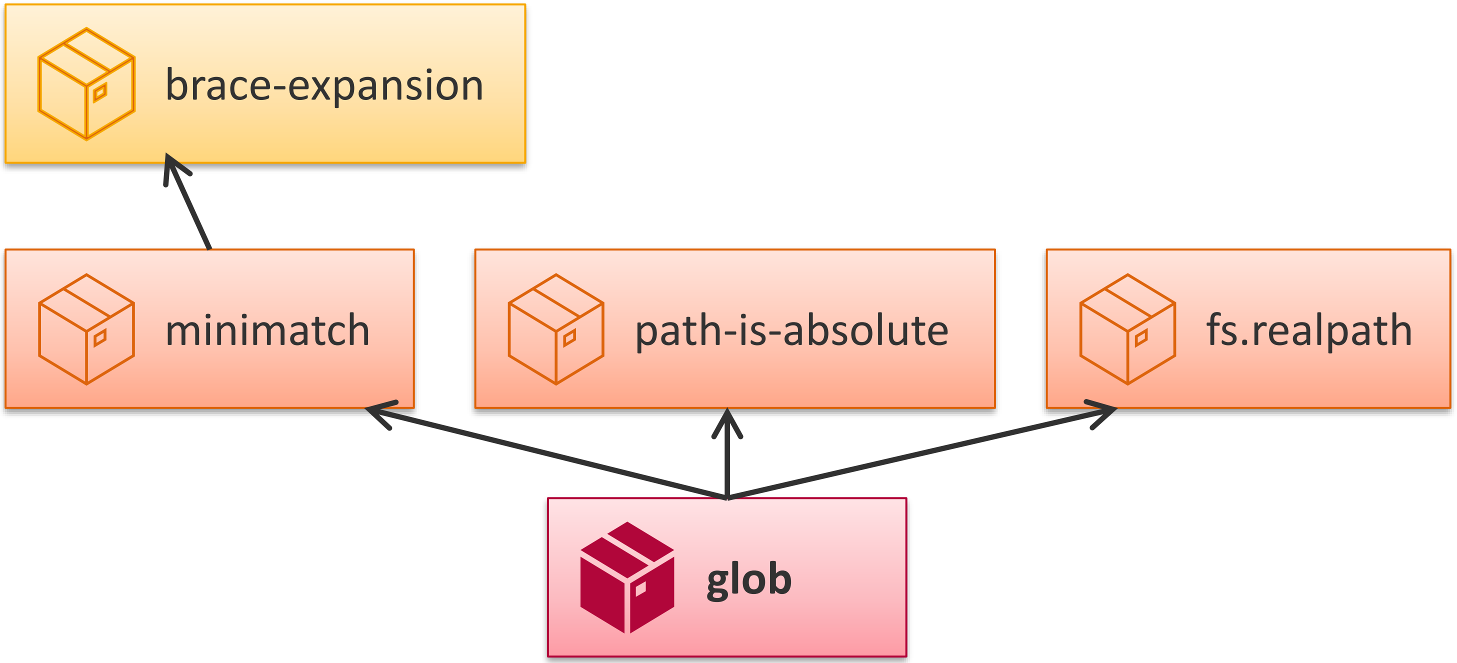
Augmenting Library Development by Mining Usage Data from Downstream Dependencies

Christoph Thiede, Willy Scheibel, Daniel Limberger, Jürgen Döllner

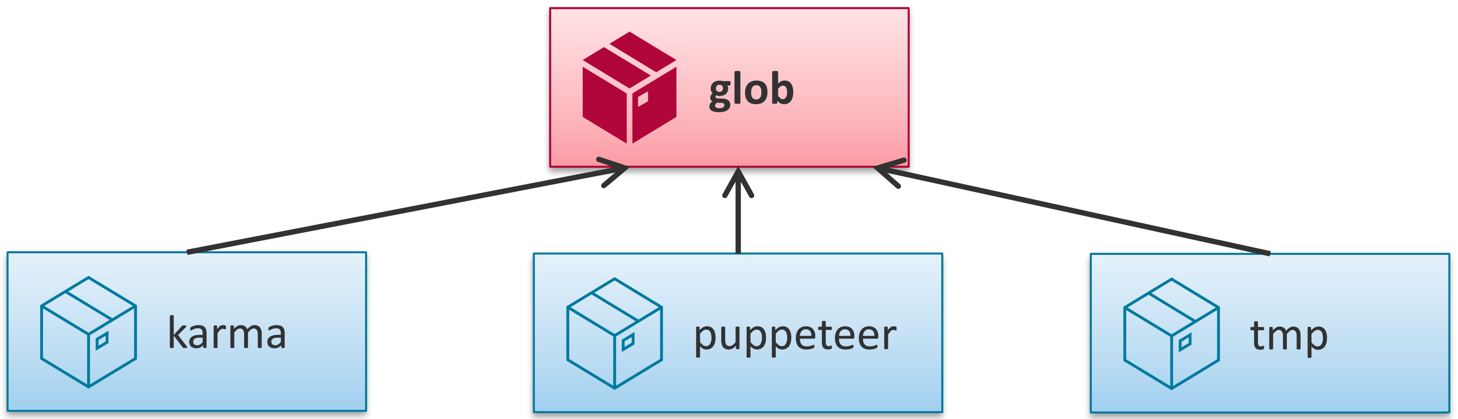
ENASE 2022

2022-04-25

Introduction: upstream dependencies



Introduction: downstream dependencies



Goal

How many dependents does my package have?

How large/important are they?

How often are certain members of my package used?

How do dependents use certain members?

How could/should we change the public interface?

Where does compatibility matter most?

practical tool

lightweight solution

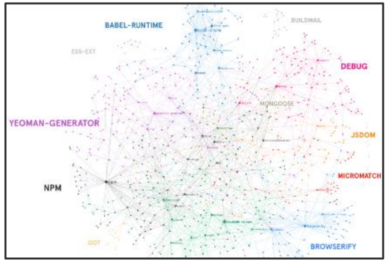
dynamically typed languages

Related work

Related work

Dependency graphs

- graph exploration [Kikas et al., 2017]
- downstream analysis (vulnerabilities) [Decan et al., 2018]



API usage analysis

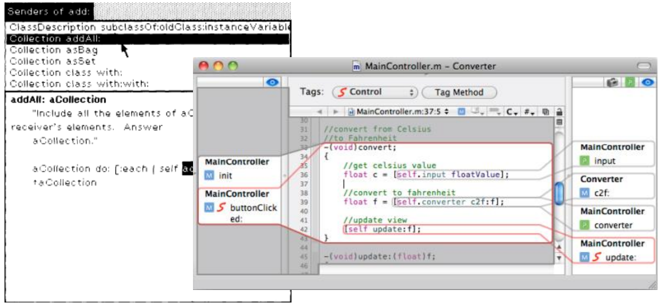
- string search [Mileva et al., 2010]
- AST scanning [Qiu et al., 2016; Sawant and Bacchelli, 2017]

Call graphs [Antal et al., 2018]

Ecosystem call graphs [Hejderup et al., 2018; Nielsen et al., 2021; Wang et al., 2020; Hejderup et al., 2021; Keshani, 2021]

Presentation

- Message Set [Goldberg, 1984]
- Stacksplorer [Karrer et al., 2011]
- Blaze [Krämer et al., 2012]
- Exapus [de Roover et al., 2013]



```
Senders of add:
ClassDescription subclassOf:oldClass:instanceVariabl
Collection addAll:
Collection asBag
Collection asSet
Collection class with:
Collection class with:with:

addAll: aCollection
    "Include all the elements of aC
receiver's elements.  Answer
    aCollection."

    aCollection do: [:each | self ad
    ↑aCollection
```

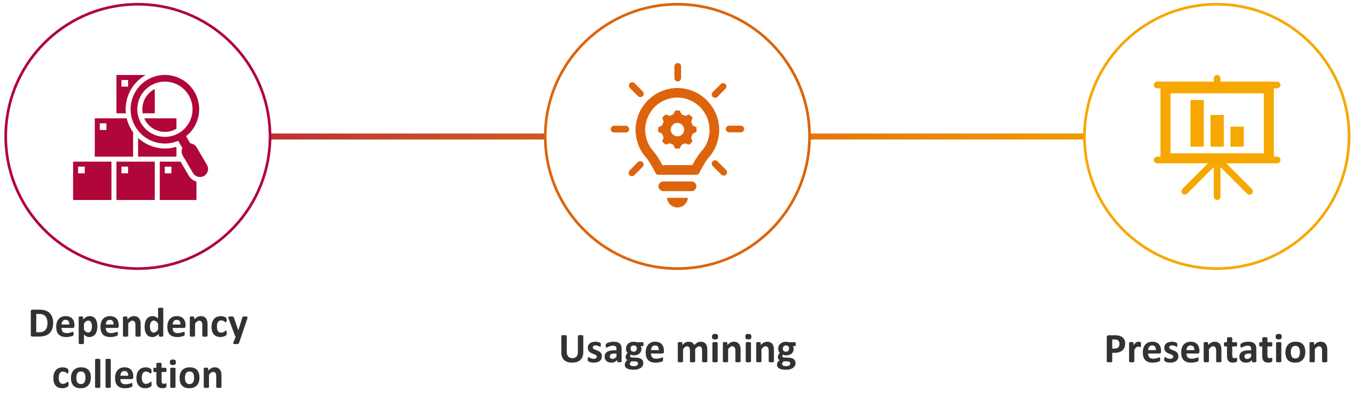
MainController.m – Converter

```
//convert from Celsius
//to Fahrenheit
-(void)convert;
{
    //get celsius value
    float c = [self.input floatValue];

    //convert to fahrenheit
    float f = [self.converter c2f:f];

    //update view
    [self update:f];
}

-(void)update:(float)f;
```

Downstream Dependency Mining

Christoph Thiede
2022-04-25

Approach



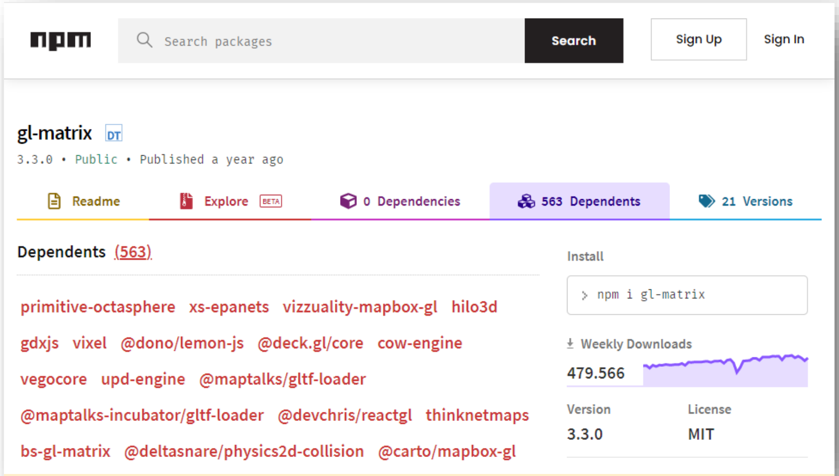
Approach
Dependency collection

- Many approaches: download large number of repos
  - Not lightweight!
- For us: pre-filter before downloading
  - Rely on already indexed/searchable cloud sources

Approach
Dependency collection

- Rely on already indexed/searchable cloud sources:

Package repositories
doubly-connected edge list of dependent packages



gl-matrix DT
3.3.0 • Public • Published a year ago

0 Dependencies | 563 Dependents | 21 Versions

Dependents (563)

primitive-octasphere xs-epanets vizzuality-mapbox-gl hilo3d
gdxjs vixel @dono/lemon-js @deck.gl/core cow-engine
vegocore upd-engine @maptalks/gltf-loader
@maptalks-incubator/gltf-loader @devchris/reactgl thinknetmaps
bs-gl-matrix @deltasnare/physics2d-collision @carto/mapbox-gl

Install

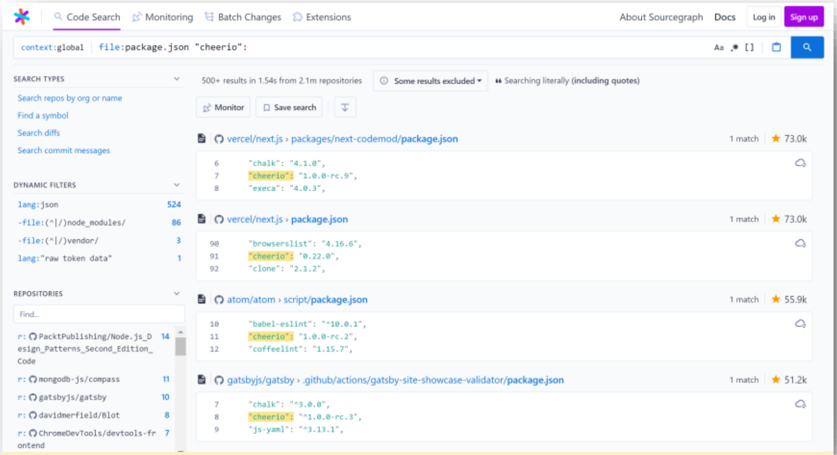
```
> npm i gl-matrix
```

Weekly Downloads
479.566

Version	License
3.3.0	MIT

npmjs.com

OSS code search engines
scan package manifest file



```
context:global  file:package.json "cheerio":
```

SEARCH TYPES
- Search repos by org or name
- Find a symbol
- Search diffs
- Search commit messages

DYNAMIC FILTERS

lang:json	524
-file:(^|/)node_modules/	86
-file:(^|/)vendor/	3
lang:"raw token data"	1

REPOSITORIES
Find...

r: PacktPublishing/Node.js_Design_Patterns_Second_Edition_Code	14
r: mongodb-js/compass	11
r: gatsbyjs/gatsby	10
r: davidmerfield/Blot	8
r: ChromeDevTools/devtools-frontend	7

500+ results in 1.54s from 2.1m repositories | Some results excluded | Searching literally (including quotes)

vercel/next.js › packages/next-codemod/package.json — 1 match ★ 73.0k

```
    "chalk": "4.1.0",
    "cheerio": "1.0.0-rc.9",
    "execa": "4.0.3",
```

vercel/next.js › package.json — 1 match ★ 73.0k

```
    "browserslist": "4.16.6",
    "cheerio": "0.22.0",
    "clone": "2.1.2",
```

atom/atom › script/package.json — 1 match ★ 55.9k

```
    "babel-eslint": "^10.0.1",
    "cheerio": "1.0.0-rc.2",
    "coffeelint": "1.15.7",
```

gatsbyjs/gatsby › .github/actions/gatsby-site-showcase-validator/package.json — 1 match ★ 51.2k

```
    "chalk": "^3.0.0",
    "cheerio": "^1.0.0-rc.3",
    "js-yaml": "^3.13.1",
```

Sourcegraph

Approach
Usage mining

```
import glob from 'glob'

glob('*.txt', { cwd: '/' }, (error, matches) => {
    if (error) {
        console.error(error)
    } else {
        console.table(matches)
    }
})
```

Approach
Usage mining

```
import glob from 'glob'

glob('*.txt', { cwd: '/' }, (error, matches) => {
    if (error) {
        console.error(error)
    } else {
        console.table(matches)
    }
})
```

Approach
Usage mining

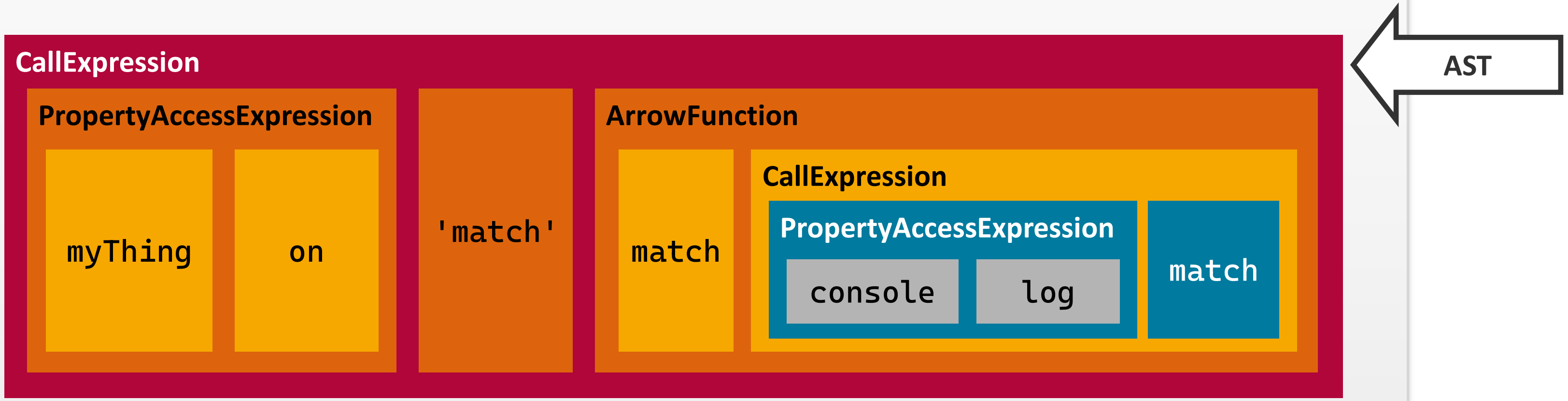
```
const myThing = new glob.Glob('*.txt')

myThing.on('match', match => console.log(match))
```

Approach
Usage mining

```
const myThing = new glob.Glob('*.txt')

myThing.on('match', match => console.log(match))
```

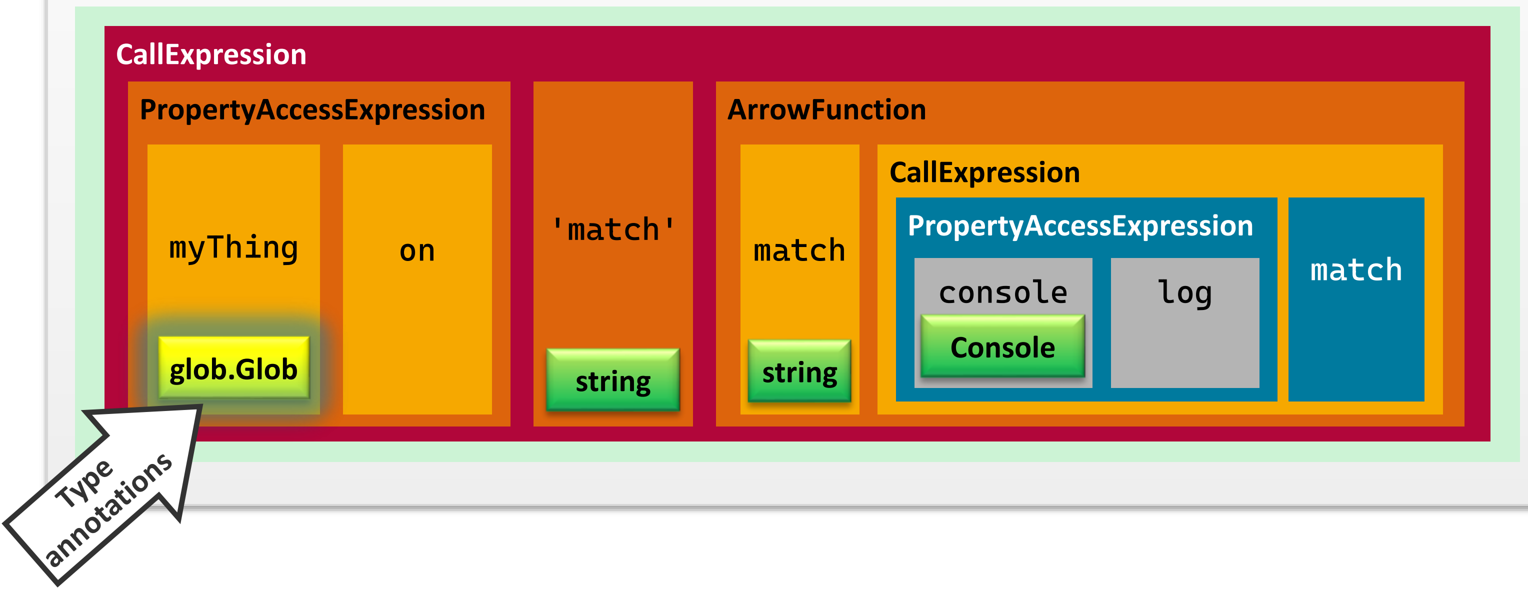
Approach
Usage mining

```
const myThing = new glob.Glob('*.txt')
```



Approach
Usage mining

```
const myThing = new glob.Glob('*.txt')
```

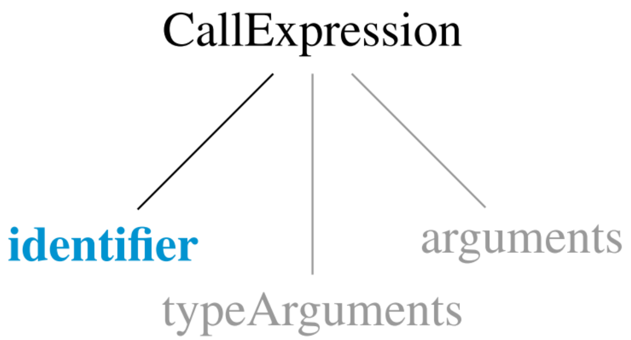


Approach
Usage mining

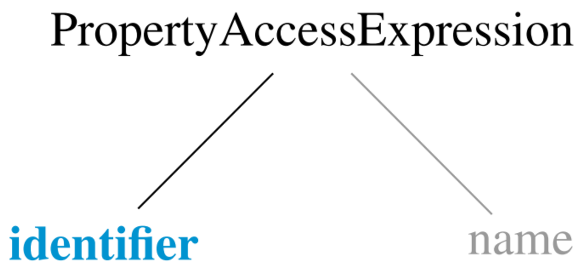
```
const myThing = new glob.Glob('*.txt')

myThing.on('match', match => console.log(match))
```

Approach

Usage mining



(a) Node pattern for a TypeScript function call, such as in: `result = fun<T1, T2>(arg1, arg2);`

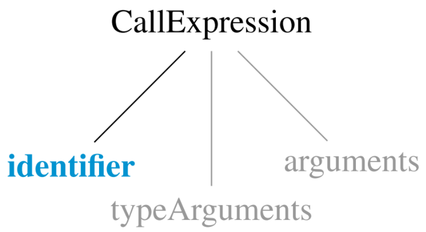


(b) Node pattern for a JavaScript property access, such as in: `return obj.prop;`

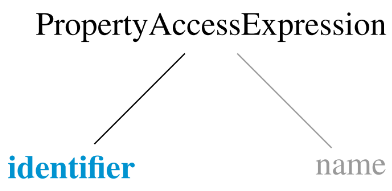
Approach
Usage mining

Input:
pkg: target package
dependencies: downstream dependencies
Output: usage samples (set of strings)

```
for $dep \in dependencies$:
    $asf \leftarrow \text{parse}(dep \cup pkg)$
    $\text{annotate\_types}(asf)$
    for $ast \in asf$:
        for $node \in dfs(ast)$:
            for $pattern \in patterns$:
                if $pattern.matches(node) \wedge pkg.declares(pattern.getType(node))$:
                    yield $node.text$
```



(a) Node pattern for a TypeScript function call, such as in: `result = fun<T1, T2>(arg1, arg2);`



(b) Node pattern for a JavaScript property access, such as in: `return obj.prop;`

Prototype

Usage mining

TypeScript

Compiler API

Presentation

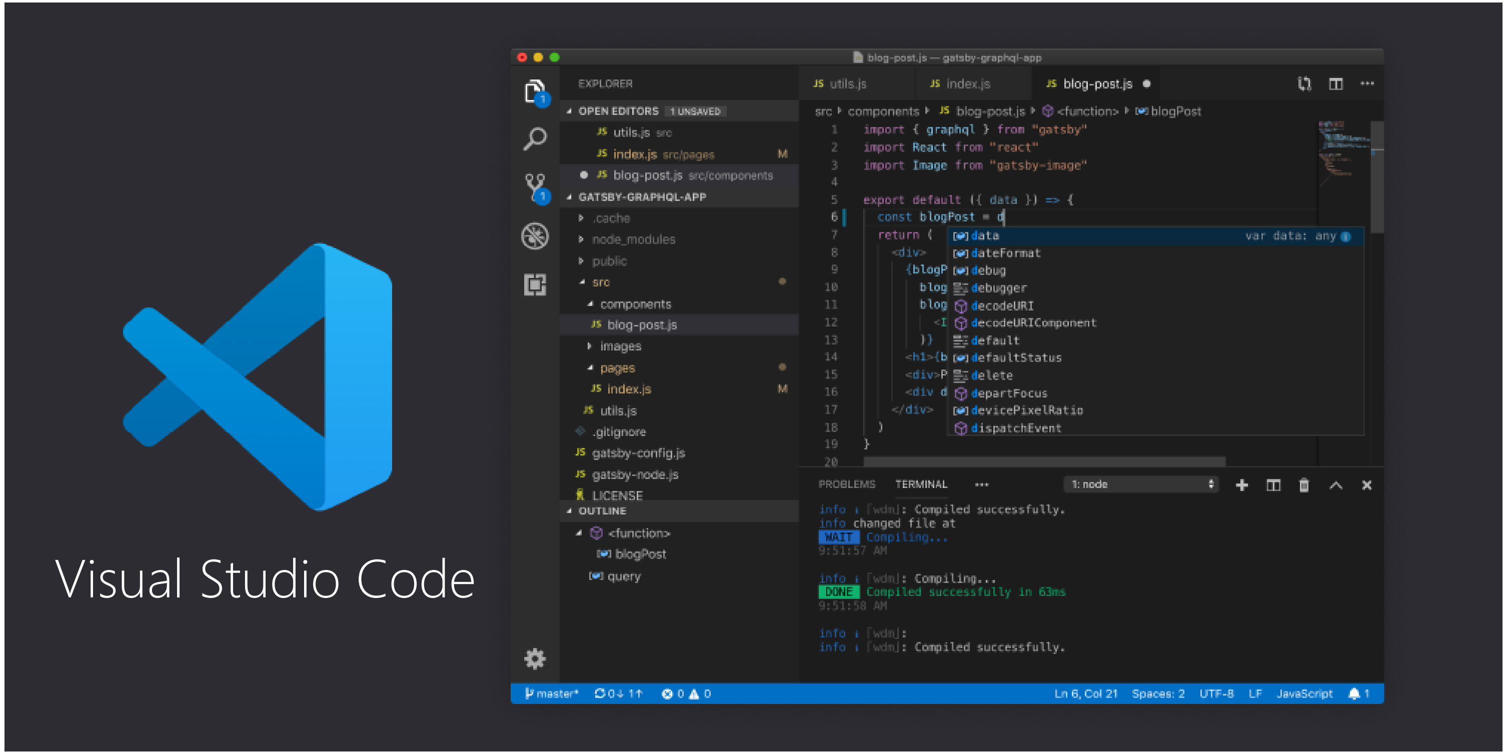
Approach
Presentation: non-functional requirements

Setup
- ready to use out of the box
- little configuration

Efficiency
- run on a single machine
- interactive response times

Integration into usual workflow

Prototype
Presentation



```
import { graphql } from "gatsby"
import React from "react"
import Image from "gatsby-image"

export default ({ data }) => {
  const blogPost = d
  return (      data                    var data: any
    <div>       dateFormat
      {blogP    debug
        blog    debugger
        blog    decodeURI
          <I    decodeURIComponent
        })      default
      <h1>{b    defaultStatus
      <div>P    delete
      <div d    departFocus
    </div>      devicePixelRatio
  )             dispatchEvent
}
```

```
info ℹ ｢wdm｣: Compiled successfully.
info changed file at
 WAIT  Compiling...
9:51:57 AM

info ℹ ｢wdm｣: Compiling...
 DONE  Compiled successfully in 63ms
9:51:58 AM

info ℹ ｢wdm｣:
info ℹ ｢wdm｣: Compiled successfully.
```

Downstream Dependency Mining

Christoph Thiede
2022-04-25

Top IDE Index (Pierre Carbonnelle, 2021): https://pypl.github.io/IDE.html

Prototype

Presentation

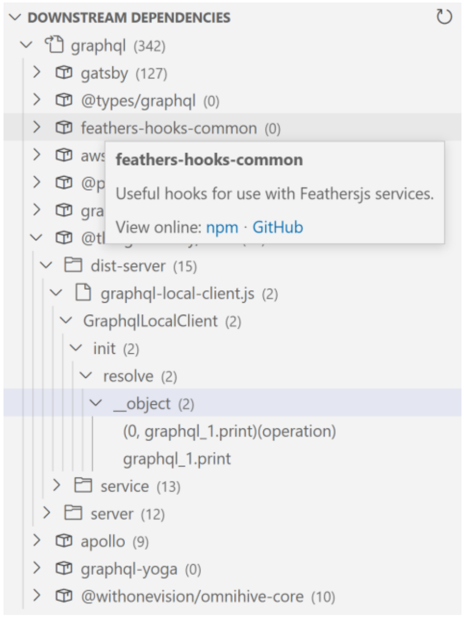
Dependency browser

Usage browser

Code annotations

Prototype

Presentation

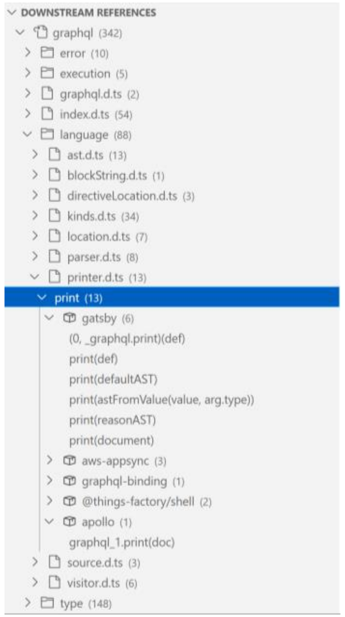


```
DOWNSTREAM DEPENDENCIES
  graphql (342)
    gatsby (127)
    @types/graphql (0)
    feathers-hooks-common (0)
    aws
    @p
    gra
    @tl
      dist-server (15)
        graphql-local-client.js (2)
          GraphqlLocalClient (2)
            init (2)
              resolve (2)
                _object (2)
                  (0, graphql_1.print)(operation)
                  graphql_1.print
        service (13)
      server (12)
    apollo (9)
    graphql-yoga (0)
    @withonevision/omnihive-core (10)
```

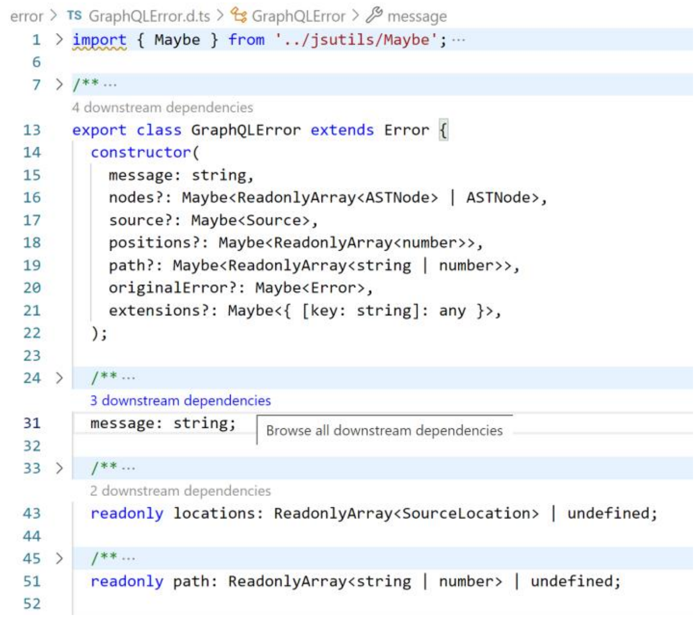
feathers-hooks-common

Useful hooks for use with Feathersjs services.

View online: npm · GitHub



```
DOWNSTREAM REFERENCES
  graphql (342)
    error (10)
    execution (5)
    graphql.d.ts (2)
    index.d.ts (54)
    language (88)
      ast.d.ts (13)
      blockString.d.ts (1)
      directiveLocation.d.ts (3)
      kinds.d.ts (34)
      location.d.ts (7)
      parser.d.ts (8)
      printer.d.ts (13)
        print (13)
          gatsby (6)
            (0, _graphql.print)(def)
            print(def)
            print(defaultAST)
            print(astFromValue(value, arg.type))
            print(reasonAST)
            print(document)
          aws-appsync (3)
          graphql-binding (1)
          @things-factory/shell (2)
          apollo (1)
            graphql_1.print(doc)
      source.d.ts (3)
      visitor.d.ts (6)
    type (148)
```



error > TS GraphQLError.d.ts > GraphQLError > message

```
import { Maybe } from '../jsutils/Maybe'; ···

/** ...
4 downstream dependencies
export class GraphQLError extends Error {
  constructor(
    message: string,
    nodes?: Maybe<ReadonlyArray<ASTNode> | ASTNode>,
    source?: Maybe<Source>,
    positions?: Maybe<ReadonlyArray<number>>,
    path?: Maybe<ReadonlyArray<string | number>>,
    originalError?: Maybe<Error>,
    extensions?: Maybe<{ [key: string]: any }>,
  );

  /** ...
  3 downstream dependencies
  message: string;      Browse all downstream dependencies

  /** ...
  2 downstream dependencies
  readonly locations: ReadonlyArray<SourceLocation> | undefined;

  /** ...
  readonly path: ReadonlyArray<string | number> | undefined;
```

Downstream Dependency Mining

Christoph Thiede
2022-04-25

Evaluation

Evaluation: research questions

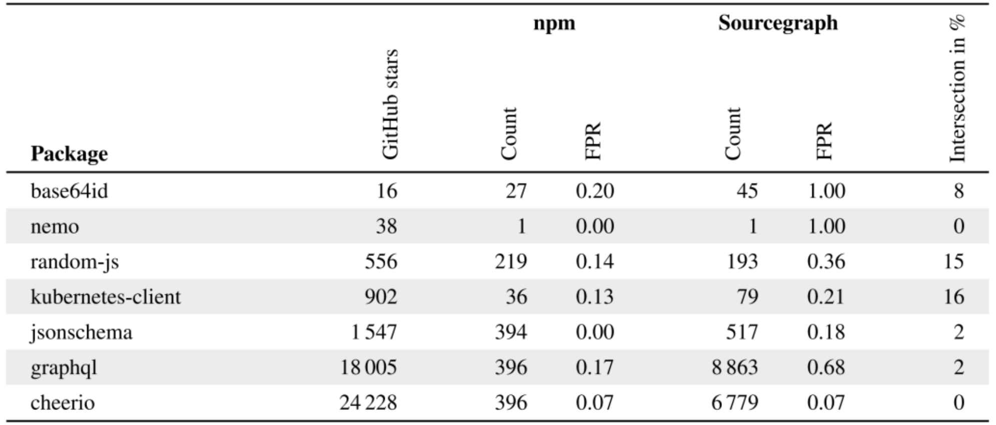
Dependency collection
(quality/quantity/
performance)

Usage sample mining
(quality/quantity/
performance)

**Applicability
of the tool**

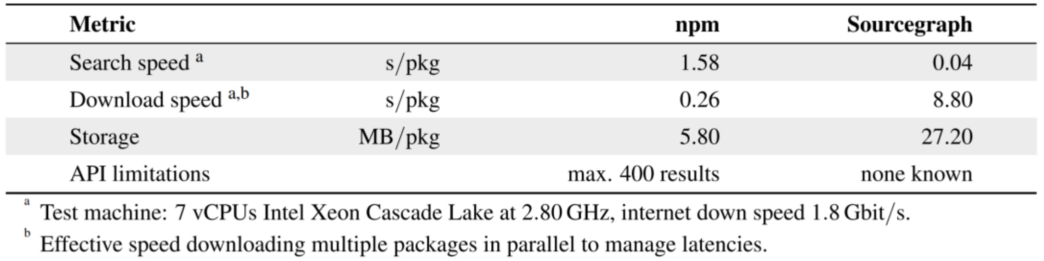
Evaluation: dependency collection

Table 1: Quantity and false-positive rates (FPR) of downstream dependencies found by the presented methods (using npm and Sourcegraph) for selected packages.



		npm		Sourcegraph		
Package	GitHub stars	Count	FPR	Count	FPR	Intersection in %
base64id	16	27	0.20	45	1.00	8
nemo	38	1	0.00	1	1.00	0
random-js	556	219	0.14	193	0.36	15
kubernetes-client	902	36	0.13	79	0.21	16
jsonschema	1 547	394	0.00	517	0.18	2
graphql	18 005	396	0.17	8 863	0.68	2
cheerio	24 228	396	0.07	6 779	0.07	0

Table 2: Performance metrics and remarks for both dependency collection methods using npm and Sourcegraph.



Metric		**npm**	**Sourcegraph**
Search speed [a]	s/pkg	1.58	0.04
Download speed [a,b]	s/pkg	0.26	8.80
Storage	MB/pkg	5.80	27.20
API limitations		max. 400 results	none known

[a] Test machine: 7 vCPUs Intel Xeon Cascade Lake at 2.80 GHz, internet down speed 1.8 Gbit/s.
[b] Effective speed downloading multiple packages in parallel to manage latencies.

- **false positives:**
  - outdated manifest files
  - peer dependencies
- **biases:**
  - invalid/missing manifest file
  - npm: only packages
  - ranking: small packages are underrepresented

Evaluation: usage mining

- **false positives:**
  - almost impossible
  - naming clashes? tricked tsc?
- **false negatives:**
  - complex build configurations (code generation, transpilers, …)
  - metaprogramming and TypeScript limitations
  - missing type definitions for intermediate frameworks
- **performance:**
  - speed*†: ~3 secs/package
  - memory*†: ~50 MB/package

```
const myThings = [new glob.Glob('*.txt')]

_.forEach(myThings, thing => thing.on(
    'match', match => console.log(match)))
```

*Sample size: 10 – 20 packages.
†Machine specs: 7 vCPUs Intel Xeon Cascade Lake @ 2.80GHz internet downspeed ~1.8 Gbit/s

Evaluation: tool

- **Non-functional requirements**
  - Setup: 10 seconds
  - Efficiency
    - Lightweight: 5 – 12 deps/min, <30 MB storage/package
    - Interactivity/temporal distance [Ungar1997]:
      - <10 seconds for first result
      - <1 second latency for navigation
  - Integration [Ungar1997]
    - spatial distance: low due to IDE extension
    - semantic distance: low due to shared artifacts
- **Answering user questions**

Future work

- **expand quantitative evaluation**
  - annotated usage samples
  - user study
- **deeper analysis of usage samples**
  - pattern mining
  - metrics
  - dynamic analysis
- **integrate further data sources**
  - change history
  - conversation platforms (issue trackers, discussion forums)
  - error stack traces in CI logs

Conclusion

How many dependents does my package have?

How large/important are they?

How often are certain members of my package used?

How do dependents use certain members?

How could/should we change the public interface?

Where does compatibility matter most?

practical tool

lightweight solution

dynamically typed languages

Dependency collection

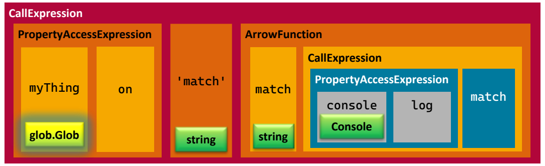
Usage mining

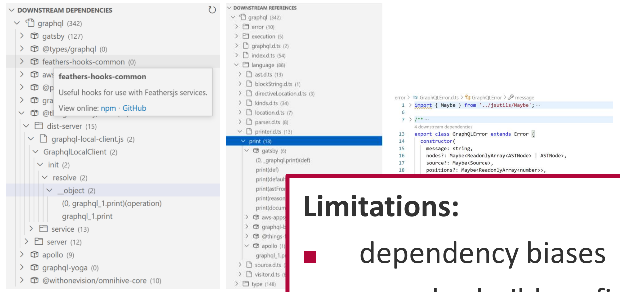
Presentation

Sourcegraph

npm

TS




```
DOWNSTREAM DEPENDENCIES
  graphql (342)
    gatsby (127)
    @types/graphql (0)
    feathers-hooks-common (0)
    aws
    @p
    gra
    @t
      dist-server (15)
        graphql-local-client.js (2)
          GraphqlLocalClient (2)
            init (2)
              resolve (2)
                _object (2)
                  (0, graphql_1.print)(operation)
                  graphql_1.print
        service (13)
      server (12)
    apollo (9)
    graphql-yoga (0)
    @withonevision/omnihive-core (10)
```

feathers-hooks-common

Useful hooks for use with Feathersjs services.

View online: npm · GitHub

```
DOWNSTREAM REFERENCES
  graphql (342)
    error (10)
    execution (5)
    graphql.d.ts (2)
    index.d.ts (54)
    language (88)
      ast.d.ts (13)
      blockString.d.ts (1)
      directiveLocation.d.ts (3)
      kinds.d.ts (34)
      location.d.ts (7)
      parser.d.ts (8)
      printer.d.ts (13)
        print (13)
          gatsby (6)
            (0, _graphql.print)(def)
            print(def)
            print(default
            print(astFro
            print(reason
            print(docum
          aws-apps
          graphql-b
          @things-
          apollo (1)
            graphql_1.p
      source.d.ts
      visitor.d.ts
    type (148)
```

```
error > GraphQLError.d.ts > GraphQLError > message
> import { Maybe } from '../jsutils/Maybe';…

> /**…
  4 downstream dependencies
  export class GraphQLError extends Error {
    constructor(
      message: string,
      nodes?: Maybe<ReadonlyArray<ASTNode> | ASTNode>,
      source?: Maybe<Source>,
      positions?: Maybe<ReadonlyArray<number>>,
```

Limitations:

- dependency biases
- complex build configurations
- ranking factors

Downstream Dependency Mining

Christoph Thiede
2022-04-25

Augmenting Library Development by Mining Usage Data from Downstream Dependencies

Christoph Thiede, Willy Scheibel, Daniel Limberger, and Jürgen Döllner

ENASE 2022

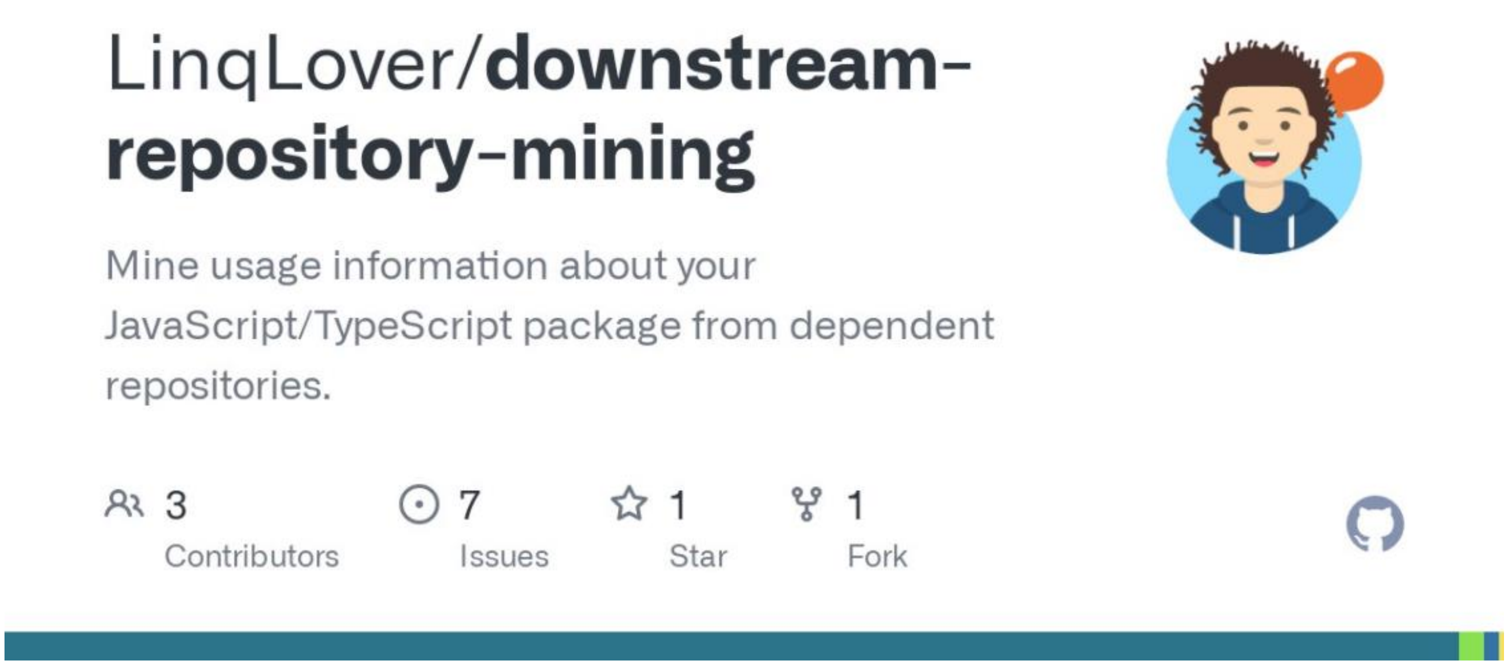
2022-04-25

Augmenting Library Development by Mining Usage Data from Downstream Dependencies

Christoph Thiede, Willy Scheibel, Daniel Limberger, and Jürgen Döllner
Hasso Plattner Institute, Digital Engineering Faculty, University of Potsdam
christoph.thiede@student.hpi.uni-potsdam.de, {willy.scheibel, daniel.limberger, juergen.doellner}@hpi.uni-potsdam.de

Keywords: Mining Software Repositories, Downstream Dependencies, API Usage.

Try it out!


LinqLover/**downstream-repository-mining**

Mine usage information about your JavaScript/TypeScript package from dependent repositories.

3 Contributors

7 Issues

1 Star

1 Fork