

Software Archaeology and Anthropology

17-313 Fall 2024

Foundations of Software Engineering

<https://cmu-17313q.github.io>

Eduardo Feo Flushing

Administrivia

- Slack
 - Please add a profile picture.
 - Ask questions in #general or #technical-questions.
 - Please use threads.
 - Use the search tool
- Office hours can be found on the course home page:
<http://cmu-17313q.github.io>

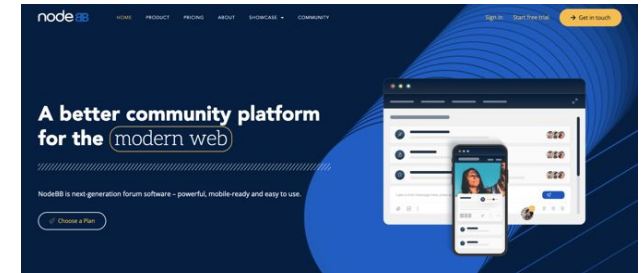
Smoking Section

- Last two rows



Team Formation Survey due Thursday

- Form groups based on schedule availability.
 - This is ridiculously important.
 - Identify experience and working styles.
 - Participation point
- Google Form, posted on slack

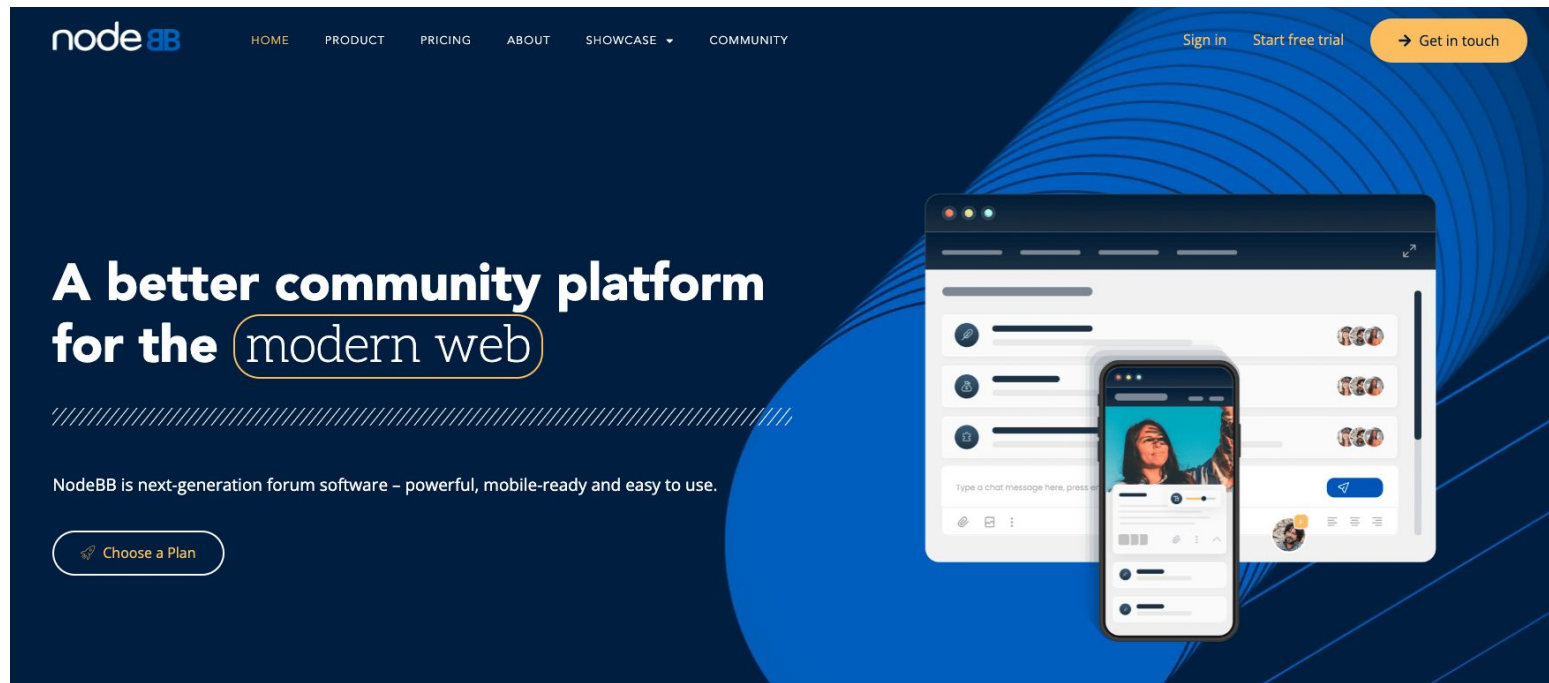


Project P1

- **P1A:** Checkpoint due next Sunday (September 1st)
 - Only 5% of total P1 points – meant to ensure you start on time
- **P1B:** Due Thursday next week, September 5th)
 - Refactor a javascript file to improve its quality
 - It will be posted tomorrow
 - Start early

Context: big old pile of code

- ... do something with it!



**You will never
understand the
entire system!**

Challenge: How do I tackle this codebase?

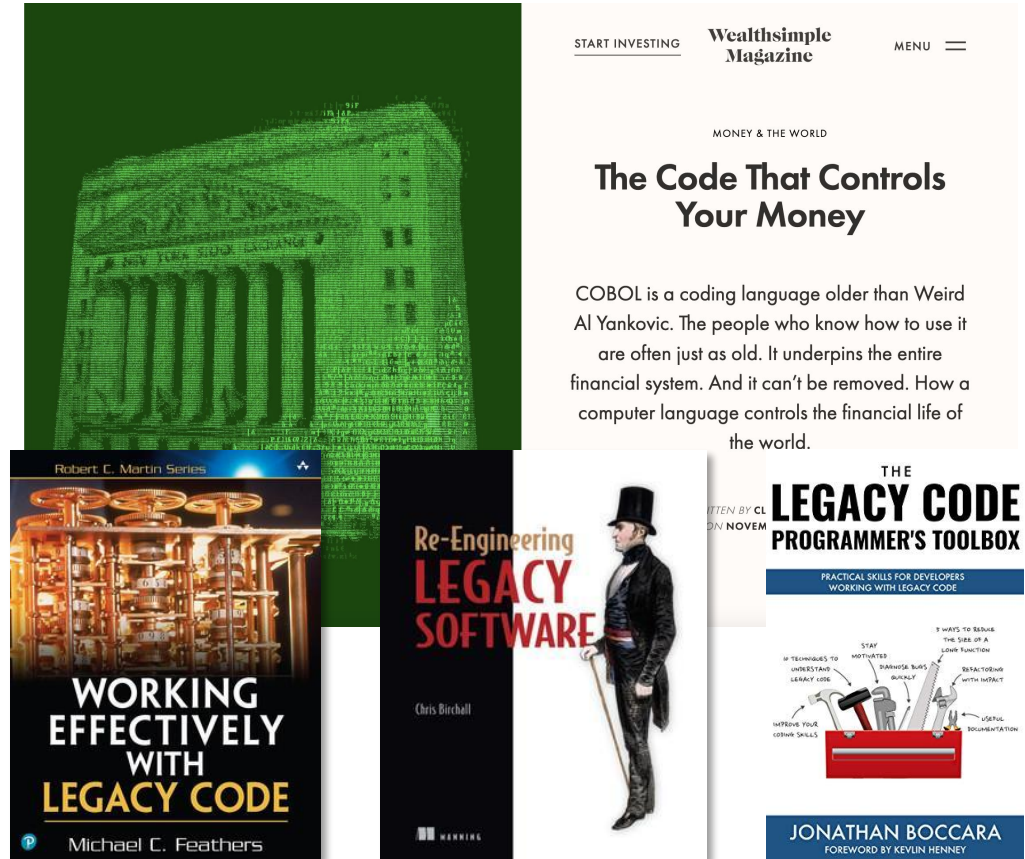


Challenge: How do I tackle this codebase?

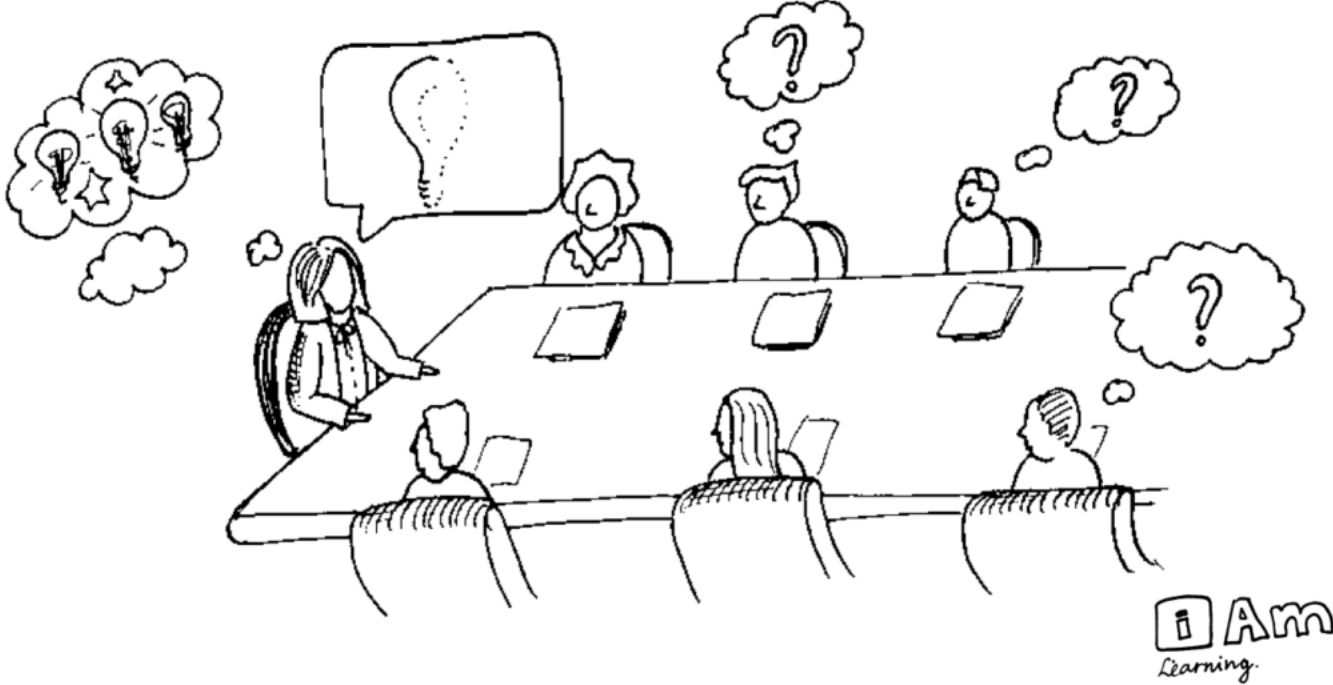
- Leverage your previous experiences (languages, technologies, patterns)
- Consult documentation, whitepapers
- Talk to experts, code owners
- Follow best practices to build a working model of the system

Bad news: There are few helpful resources!

- **Working Effectively with Legacy Code.**
Michael C. Feathers. 2004.
- **Re-Engineering Legacy Software.**
Chris Birchall. 2016.
- **The Legacy Code Programmer's Toolbox.**
Jonathan Boccara. 2019.



Why? Because of Tacit Knowledge



Today: How to tackle codebases

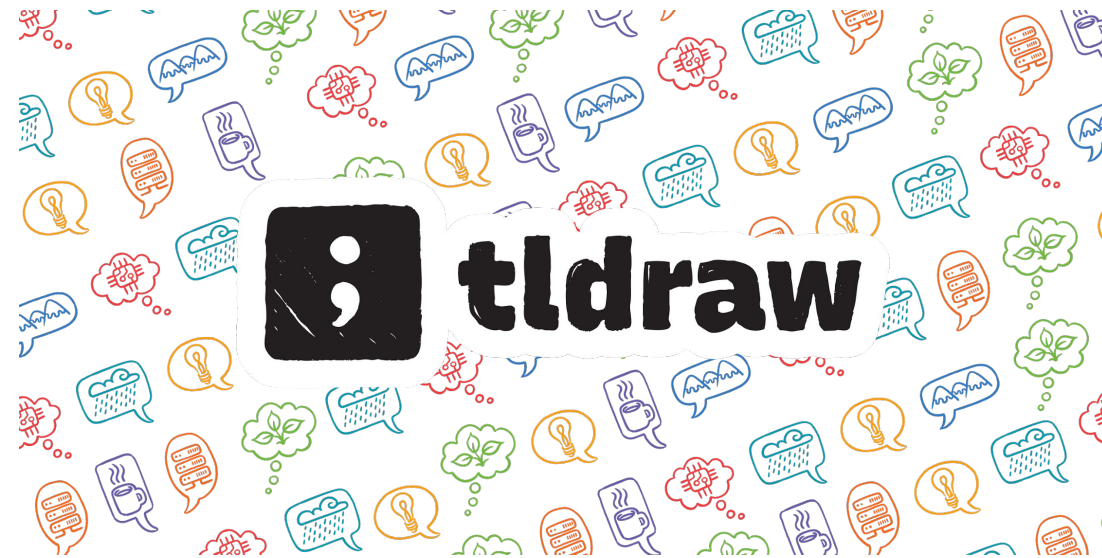
- Goal: develop and test a working model or set of working hypotheses about how (some part of) a system works
- Working model: an understanding of the pieces of the system (components), and the way they interact (connections)
- Focus: Observation, probes, and hypothesis testing
 - Helpful tools and techniques!



essentially,
all models are wrong,
but some are useful

George E. P. Box

Live Demonstration: tldraw



<https://github.com/tldraw/tldraw>

Steps to Understand a New Codebase

- Look at README.md
- Clone the repo.
- Build the codebase.
- Figure out how to make it run.
- What do you want to mess with?
 - Clone and own
- Traceability - Attach a debugger
 - View Source
 - Find the logs.
 - Search for constants (strings, colors, weird integers (#DEADBEEF))

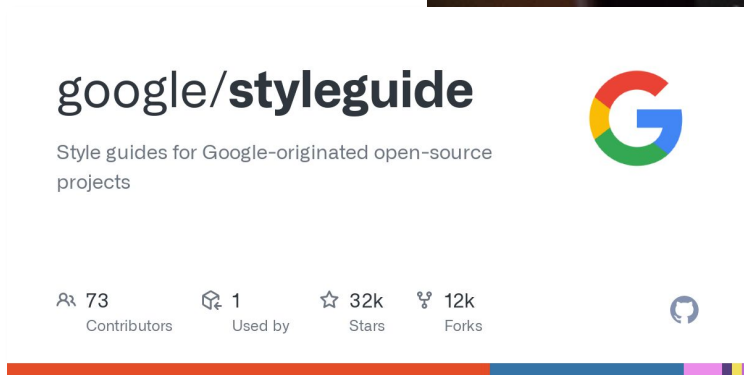
Participation Activity

- Take out a piece of paper.
- Write down one pro and one con about trying to understand a new codebase by compiling and building it vs. just reading the code.
- Pair with your neighbor and discuss your answers. Do you agree?
- Share with the class!
- Write your own andrewID on the paper, leave it at the end of class.

Observation: Software is full of patterns

- File structure
- System architecture
- Code structure
- Names
- ...

```
31 def __init__(self, *args, **kwargs):
32     self.file = None
33     self.fingerprints = set()
34     self.logdupes = True
35     self.debug = debug
36     self.logger = logging.getLogger(__name__)
37     if path:
38         self.file = open(os.path.join(path, "requests.json"),
39                          "a")
40         self.file.seek(0)
41         self.fingerprints.update(e.request for e in self.requests)
42
43 @classmethod
44 def from_settings(cls, settings):
45     debug = settings.getbool("SUPERFINGERPRINT")
46     return cls(job_dir(settings), debug)
47
48 def request_seen(self, request):
49     fp = self.request_fingerprint(request)
50     if fp in self.fingerprints:
51         return True
52     self.fingerprints.add(fp)
53     if self.file:
54         self.file.write(fp + os.linesep)
55
56 def request_fingerprint(self, request):
57     return request_fingerprint(request)
```



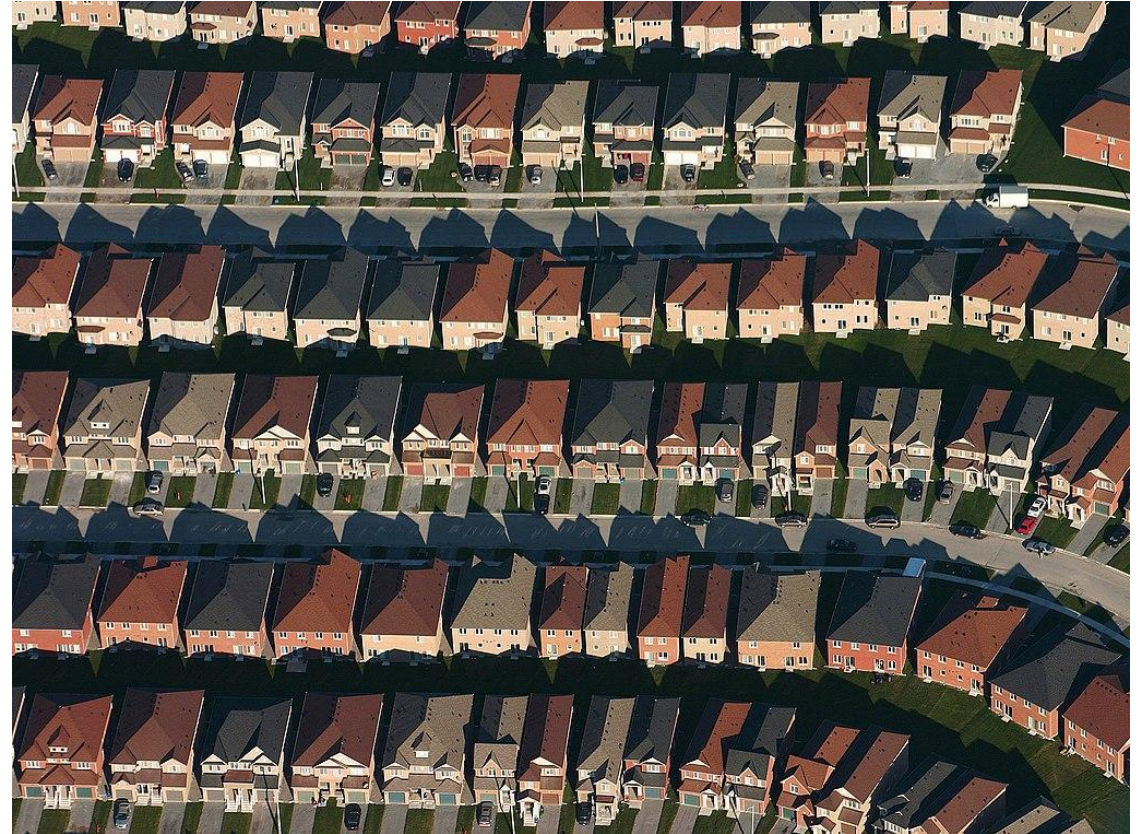
google/styleguide

Style guides for Google-originated open-source projects

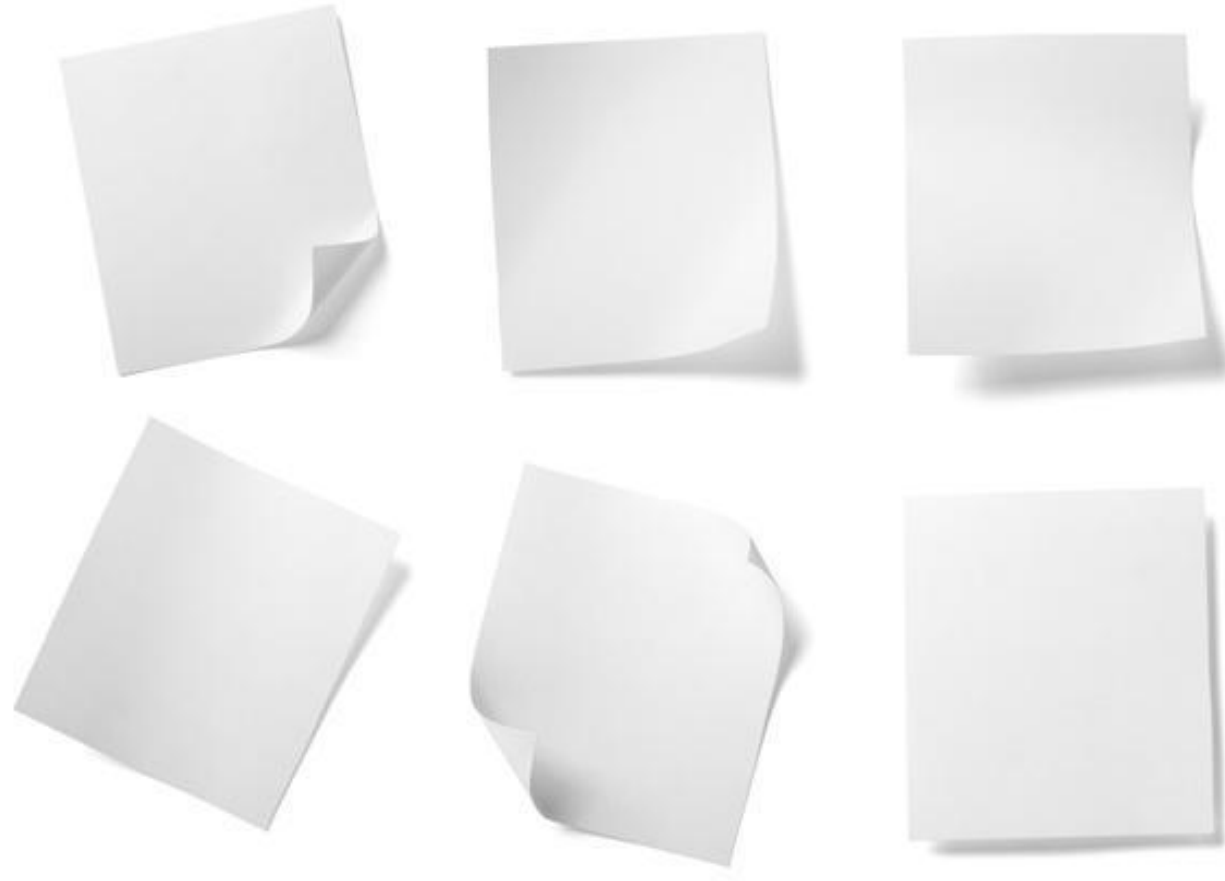
73 Contributors 1 Used by 32k Stars 12k Forks

Observation: Software is massively redundant

- There's always something to copy/use as a starting point!



Observation: Code must run to do stuff!



Observation: If code runs, it must have a beginning...



The Beginning: Entry Points

- Locally installed programs: run cmd, OS launch, I/O events, etc.
- Web apps server-side: Browser sends HTTP request (GET/POST)
- Web apps client-side: Browser runs JavaScript, event handlers

Can running code be Probed/Understood/Edited?

Transparent



Source code built locally

(P+U+E)

Translucent



Binaries running locally

Open source

(P+U)

Closed source

(P)

Opaque



Server-side apps running remotely

Open source

(U)

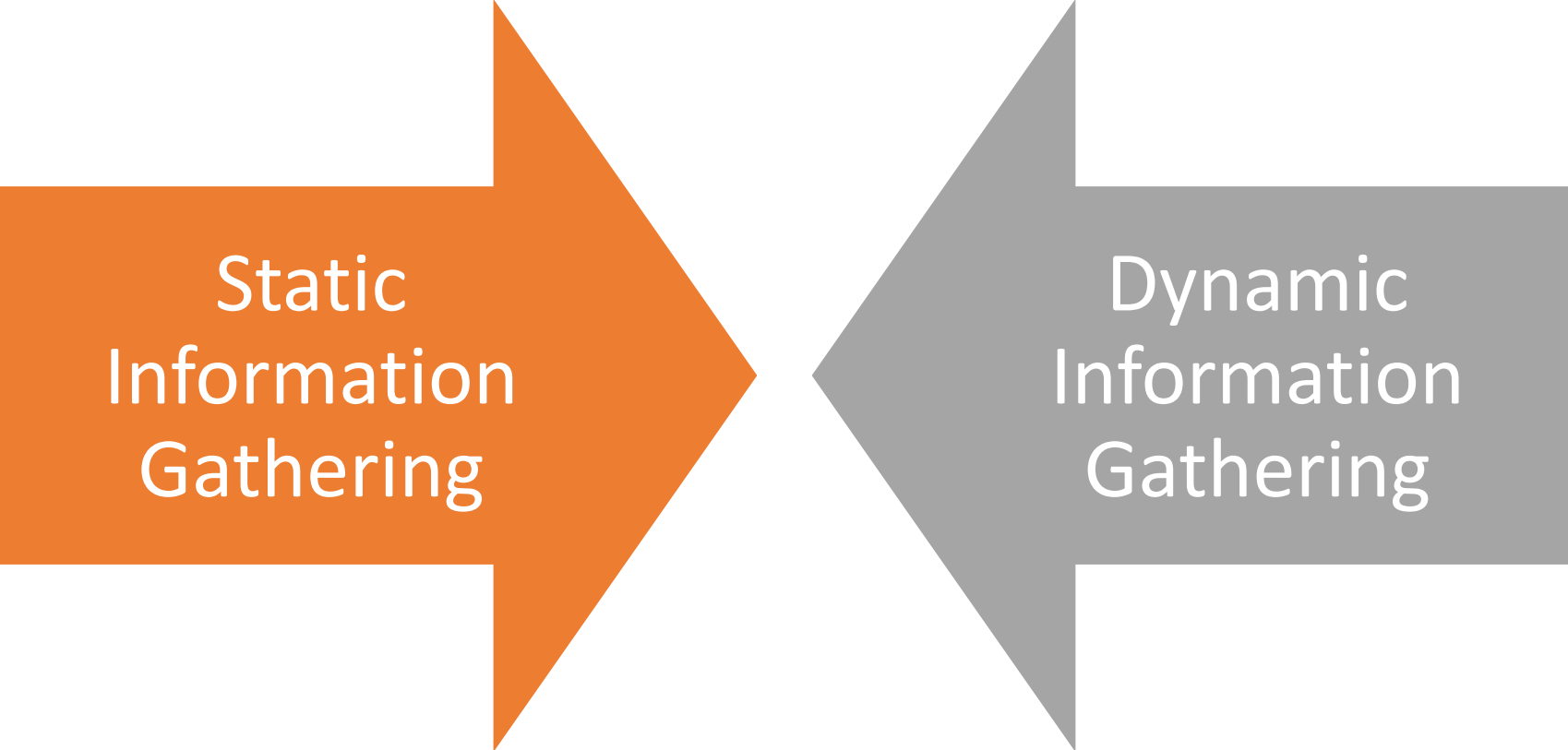
Closed source

(Talk to NSA)

Creating a model of unfamiliar code



Source code built
locally

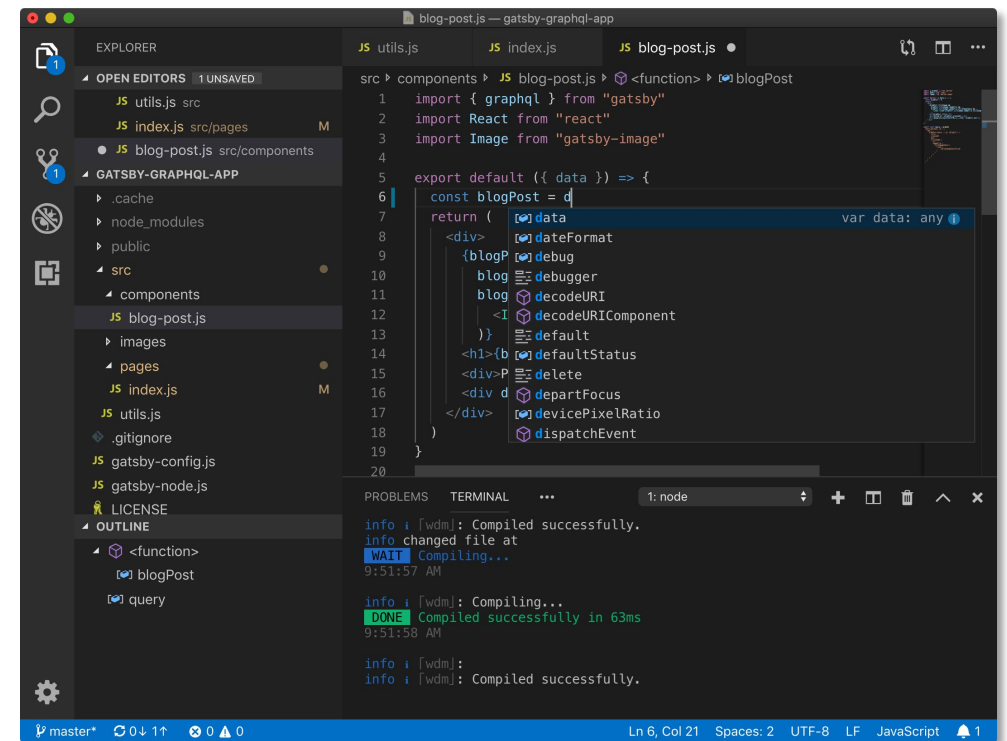
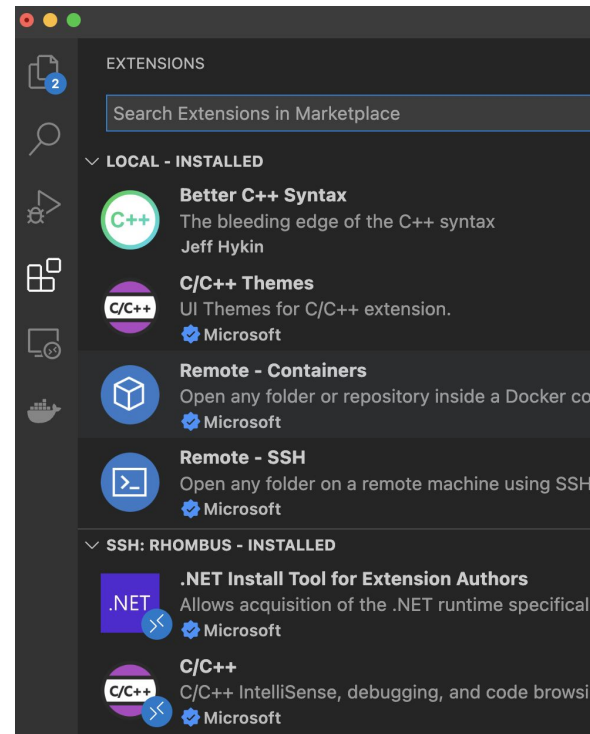


Static Information Gathering

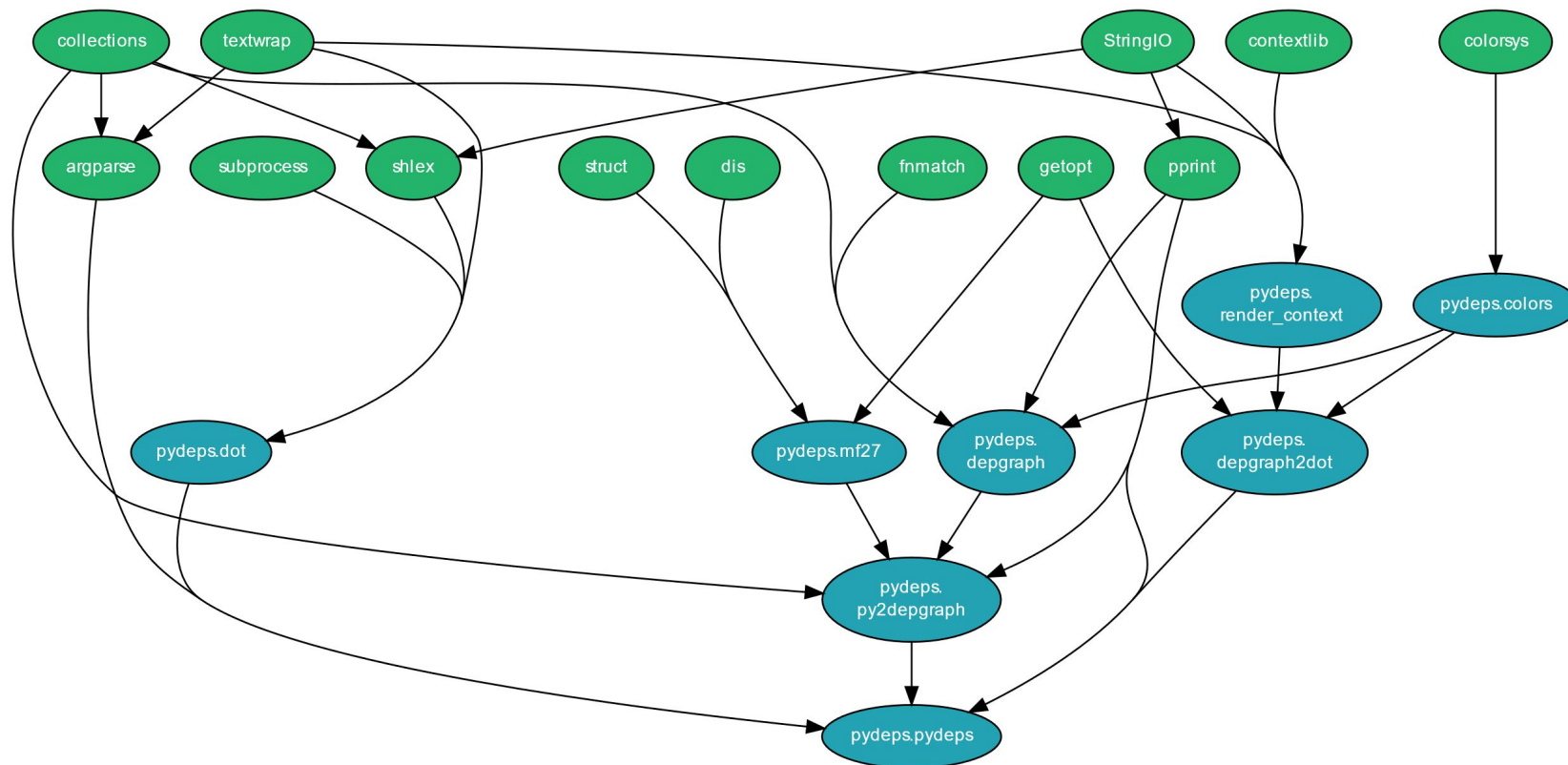
- Basic needs:
 - Code/file search and navigation
 - Code editing (probes)
 - Execution of code, tests
 - Observation of output (observation)
- Many choices here on tools! Depends on circumstance.
 - grep/find/etc. Knowing Unix tools is invaluable
 - A decent IDE
 - Debugger
 - Test frameworks + coverage reports
 - Google (or your favorite web search engine)
 - ChatGPT or LaMA

Static Information Gathering: Use an IDE!

Real software is too complex to keep in your head

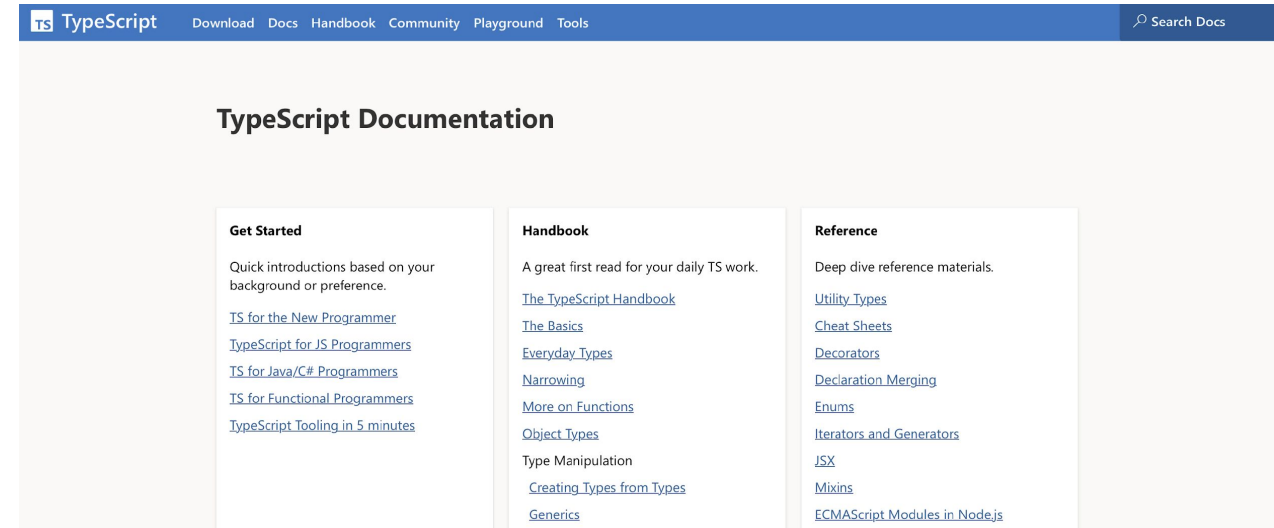


Dependency maps



Consider documentation and tutorials judiciously

- Great for discovering entry points!
- Can teach you about general structure, architecture (more on this later in the semester)
- Often out of date.
- As you gain experience, you will recognize more of these, and you will immediately know something about how the program works
- Also: discussion boards; issue trackers



The screenshot shows the TypeScript Documentation website. The header includes the TypeScript logo and navigation links: Download, Docs, Handbook, Community, Playground, Tools, and a search bar labeled 'Search Docs'. The main content area is titled 'TypeScript Documentation' and is divided into three columns:

- Get Started**: Quick introductions based on your background or preference. Links include: [TS for the New Programmer](#), [TypeScript for JS Programmers](#), [TS for Java/C# Programmers](#), [TS for Functional Programmers](#), and [TypeScript Tooling in 5 minutes](#).
- Handbook**: A great first read for your daily TS work. Links include: [The TypeScript Handbook](#), [The Basics](#), [Everyday Types](#), [Narrowing](#), [More on Functions](#), [Object Types](#), [Type Manipulation](#), [Creating Types from Types](#), and [Generics](#).
- Reference**: Deep dive reference materials. Links include: [Utility Types](#), [Cheat Sheets](#), [Decorators](#), [Declaration Merging](#), [Enums](#), [Iterators and Generators](#), [JSX](#), [Mixins](#), and [ECMAScript Modules in Node.js](#).

Discussion Boards and Issue Trackers

The screenshot shows the Stack Overflow search results page for the query "java on mac". The page includes a navigation bar with "Home", "About", "Products", and "For Teams". The search bar contains "java on mac" and there are "Log in" and "Sign up" buttons. The search results are displayed in a list format, with the top result being "How to set or change the default Java (JDK) version on mac-OS?". This result has 1311 votes, 36 answers, and 1.4m views. Other results include "How to install Java 8 on Mac" (1271 votes, 34 answers, 1.3m views) and "Where is Java Installed on Mac OS X?" (861 votes, 20 answers, 1.0m views). A sidebar on the left contains navigation links for "Home", "Public", "Questions", "Tags", "Users", "Companies", "Collectives", and "Teams". A large advertisement for Stack Overflow for Teams is visible on the right side of the page.

The screenshot shows the GitHub Issues page for the repository "sismics / reader". The page includes a navigation bar with "Code", "Issues", "Pull requests", "Actions", "Projects", "Wiki", "Security", and "Insights". The search bar contains "Type to search". The issues are displayed in a list format, with the top issue being "Rss feed" (#182, opened 3 weeks ago by TeckboyAJ). Other issues include "Docker and database docker name" (#181, opened on Nov 2, 2022 by Merrick28), "Error on OPML import" (#177, opened on Mar 14, 2021 by asm0dey), "feature request: naive bayes ham / spam classifier" (#176, opened on Nov 11, 2020 by ag88), "file is broken msg on mac" (#175, opened on Sep 13, 2020 by ksdavidc), "default credentials don't work" (#174, opened on Sep 8, 2020 by dowodenum), "Detect duplicate article in different feed" (#169, opened on Dec 23, 2019 by cloutierjo), and "Android : Dark mode" (#167, opened on Nov 12, 2018 by jendib). A sidebar on the left contains navigation links for "Code", "Issues", "Pull requests", "Actions", "Projects", "Wiki", "Security", and "Insights".

Dynamic Information Gathering

Change helps to inform and refine mental models

- Build it.
- Run it.
- Change it.
- Run it again.
- How did the behavior change?

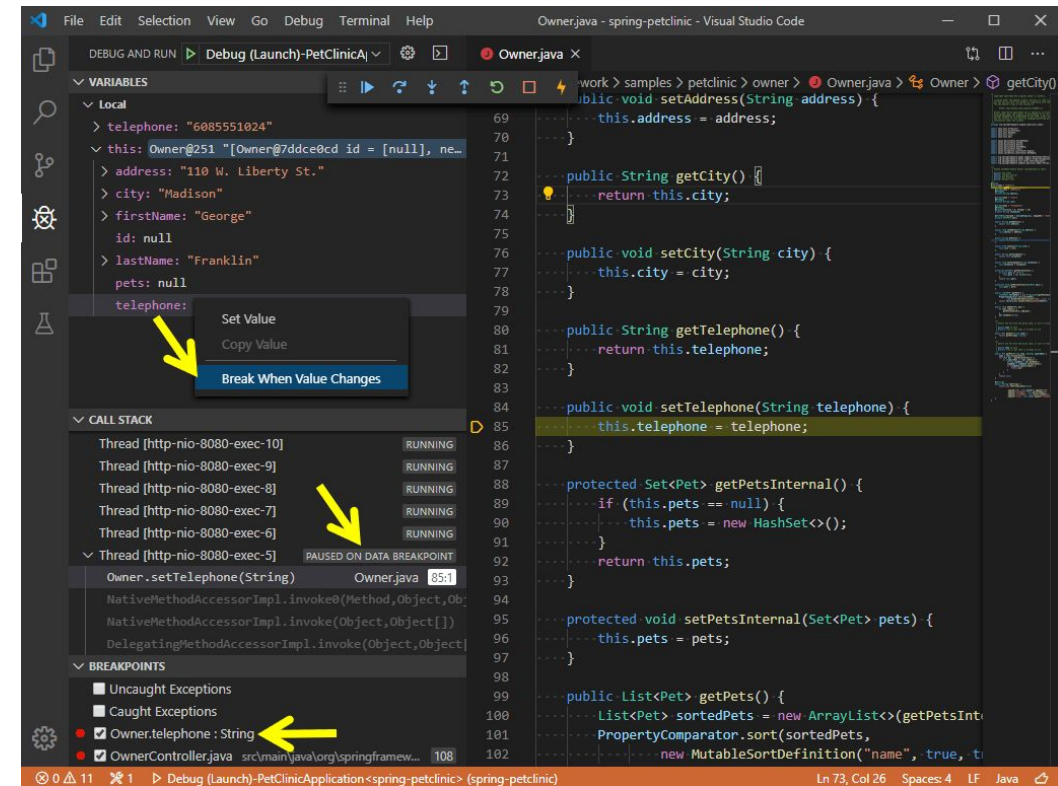


How to start?

- Confirm that you can build and run the code.
 - Ideally both using the tests provided, and by hand.
- **Confirm that the code you are running is the code you built!**
- Confirm that you can make an externally visible change
- How? Where? Starting points:
 - Run an existing test, change it
 - Write a new test
 - Change the code, write or rerun a test that should notice the change
- Ask someone for help

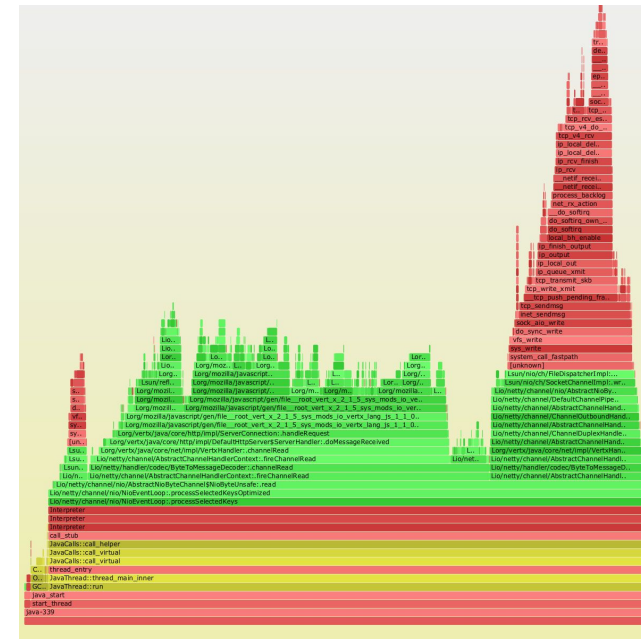
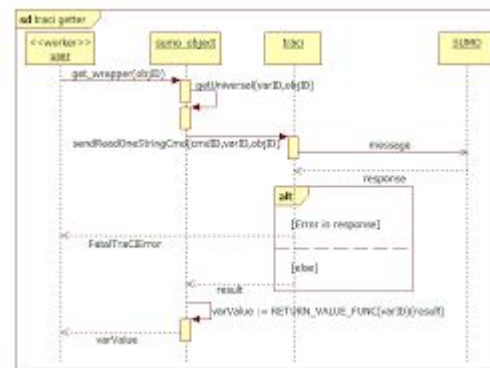
Probes: Observe, control or “lightly” manipulate execution

- `print(“this code is running!”)`
- Structured logging
- Debuggers
 - Breakpoint, eval, step through / step over
 - (Some tools even support remote debugging)
- Delete debugging
- Chrome Developer Tools



Runtime code analysis tools

- Collect runtime traces and visualize them
 - Flame graphs
 - Sequence diagrams
- Use judiciously



Tip: Find a particular thing and trace the action backward

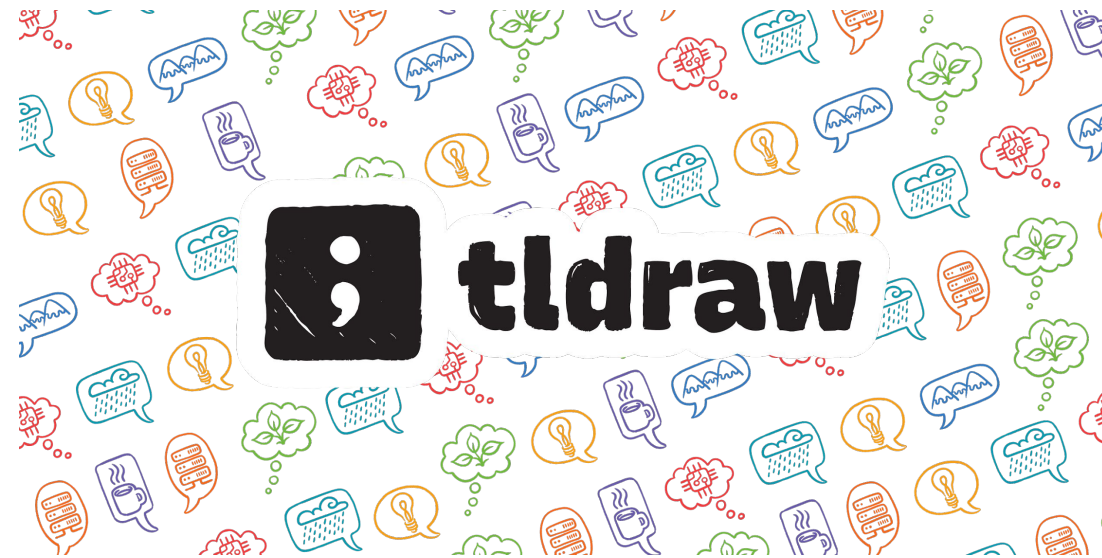
The screenshot shows the NodeBB interface with the 'Categories' page. The 'Announcements' category is circled in black. A blue arrow points from this category to a post in the 'General Discussion' category. The post is by user '@admin' and says 'Hello', posted 'about 16 hours ago'. The table below shows the statistics for each category.

Category	TOPICS	POSTS	Recent Post
Announcements	0	0	No new posts.
General Discussion	1	2	@admin Hello (about 16 hours ago)
Comments & Feedback	0	0	No new posts.
Blogs	0	0	No new posts.

Powered by NodeBB | Contributors

E.g.,
Where do categories come from?
How are they stored?
How are they rendered?

Let's try some of these techniques again...



<https://github.com/tldraw/tldraw>

Remember...

- Reading and understanding code is one of the most important skills you should learn
- It's common to get stuck or feel overwhelmed. **Don't give up!**
- Consider yourself lucky! Things are much easier today



Learning Goals

- Understand and scope the task of taking on and understanding a new and complex piece of existing software
- Appreciate the importance of configuring an effective IDE
- Contrast different types of code execution environments including local, remote, application, and libraries
- Enumerate both static and dynamic strategies for understanding and modifying a new codebase