CS130: Software Engineering

Lecture 3: Code Reviews, Web Servers

https://forms.gle/VebFxiAQxDQVju5S6

A word: How was the assignment?

A tweet: What makes a code review good?

A vote: Config parser should accept or reject empty input?

Assignment 1 Postmortem

Assignment 1 ~~Postmortem~~ Retrospective

What went well?

What did not go so well?

- Windows?
- Bash?
- git / Gerrit workflow
- Debugging problems on Piazza
- Code reviews take time

Empty config

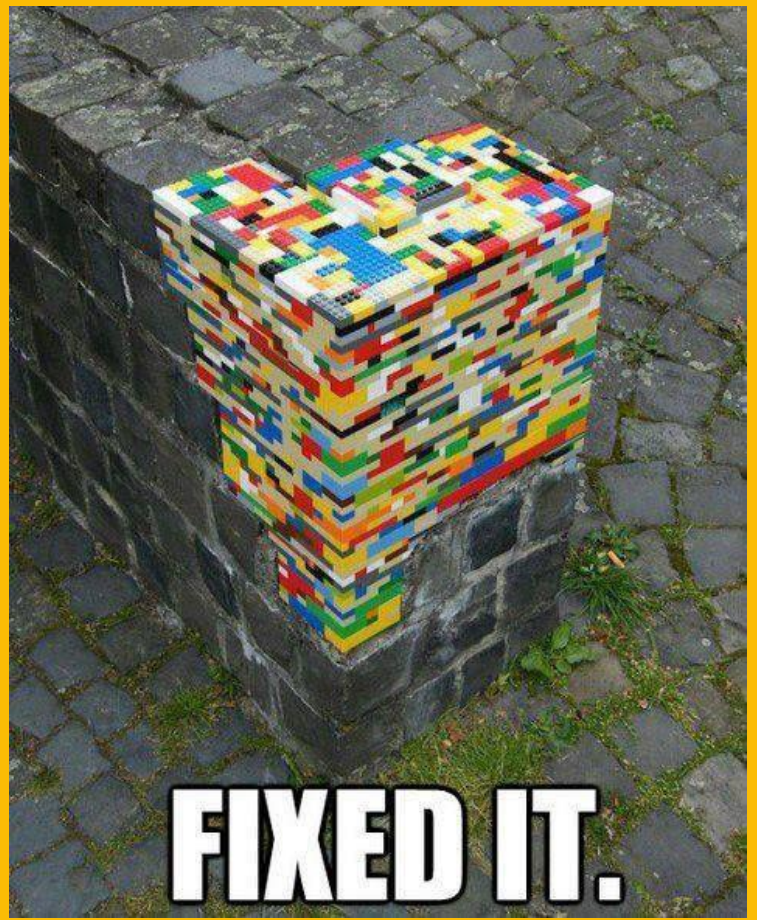
What should your parser do on an empty config?

Blockers

- Any step in your weekly work that is required before you can do any of the future steps
- As an IC (Individual Contributor, i.e. not the TL)
  - After you have decided what piece of the assignment you are responsible for, immediately think through the whole chunk and identify points where you may get blocked.
  - Make sure you get through those sections early in the week!
- As a TL
  - You’re #1 priority every day is making sure your team is not blocked
  - Code reviews, effective planning at weekly meeting, etc.

Other things

- Ask questions effectively!
- Assignments are intentionally open-ended...
- Everything has Pros and Cons!
- Not everything given to you will be perfect, improve it!
- Practice, practice, practice
- “Safe” and “risky” actions



Lecture 2 followup: Types of tests

Types of tests

- Unit tests
- Integration tests
- Regression tests
- Canaries
- Probers
- Load tests
- Fuzz tests
- … and more

Unit or Integration?

Unit tests

- Test a single class ("unit")
- Written in the same language as the code under test.

Integration Tests

- Test the whole binary or system
- Often written in a scripting language, like Bash or Python.

Integration test for a web server

Components:

- A config file
- A file containing the expected output
- A shell script to automate everything

Integration test for a web server

```
#!/bin/bash

set -e  # Abort if any command fails
make httpserver

./httpserver testdata/config &>/dev/null &  # Run silently in bg
curl -s -I http://localhost:12345 > /tmp/actual  # Make a request
kill $!  # Shut down the server.

diff testdata/expected /tmp/actual  # Exit code is 1 if different
```

Other types of tests: Regression tests

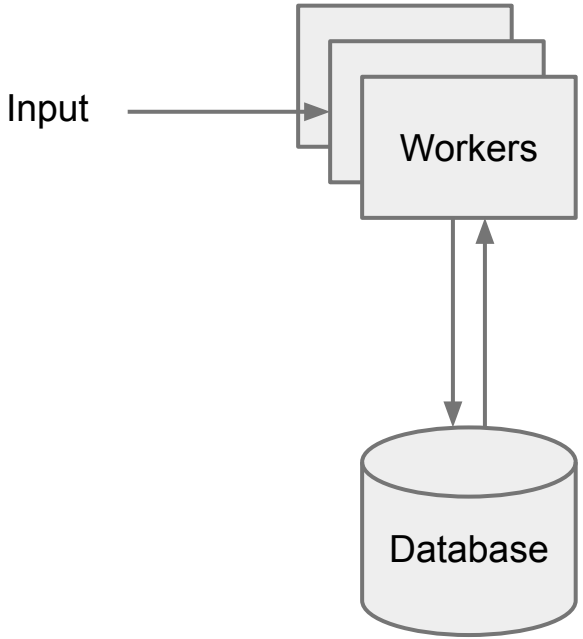
Example: a web indexing system

- Input: web pages
- Output: web index
- Choose some web pages (inputs)
- Run them through the system, record the outputs ("goldens")
- Make changes, run again
- Compare the new outputs to the goldens.
- If you're happy with the changes, replace the goldens with the new outputs

Other types of tests: Canaries

Example: A database and a worker

- Run a new worker in read-only mode (a canary)
- Does everything except write to the database.
- Does it crash?
- Does it use more memory or CPU than before?
- Does it log any new errors?



Other types of tests: Probers

- Test your full production system

Example: Google web search

- Every minute, search for "facebook"
- Is the top result facebook.com?
- If not, page someone!

Other types of tests: Load tests

- How does the system handle heavy traffic?

Example: your web server

- Run your webserver
- Generate artificial load (say, 100,000 requests per second)
- Does it slow down?
- Does it crash?
- Does it return errors?

Other types of tests: Load testing, a personal story

USA TODAY

Home | News | Travel | Money | Sports | Life | Tech | V

Sports » Olympics

Overwhelming demand for Beijing Olympics tickets crashes computer network

Posted 10/30/2007 7:06 AM | Comment | Recommend

E-mail | Print |

BEIJING (AP) — Tickets to the Beijing Olympics went on sale in China on a first-come, first-served basis Tuesday, but overwhelming demand crashed the computer ticketing system.

More than eight hours after ticket sales began, the online ticketing site had a note saying the system was busy and to check back later. The telephone ticket hot line rang busy, as it had all day.

"The speed of the ticketing system is relatively slow," the Beijing Olympics organizing committee (BOCOG) said in an announcement on its Web site. "It is temporarily unavailable ... those who want to buy tickets through Bank of China branches or the telephone hot line need to try again later."

The announcement did not mention the online ticketing site. However, a ticketing department official said in a statement last week that a main computer system processed all transactions.

Other types of tests: Fuzz Testing

- Testing a system with random inputs

Example: Config parser

- Generate a random string, try to parse it.
- Does the parser crash? Does it misbehave?
- For repeatability, store the random seed.

Code Reviews

Main idea

Source code isn't valuable unless someone else can use it!

Code reviews

Best proof of reusability is review.

For code intended for reuse:

- Give it to someone else and see if they appreciate it.
- They should be able to merge it into the stack and use it.

Code review ensures that someone else values and understands the code.

Quick question

Consider the following, think quickly:

A hammer and a nail cost $1.10. The hammer costs $1 more than the nail.

How much does the hammer cost?

$2.10?

$1.10?

$1?

$1.05!

Thinking systems

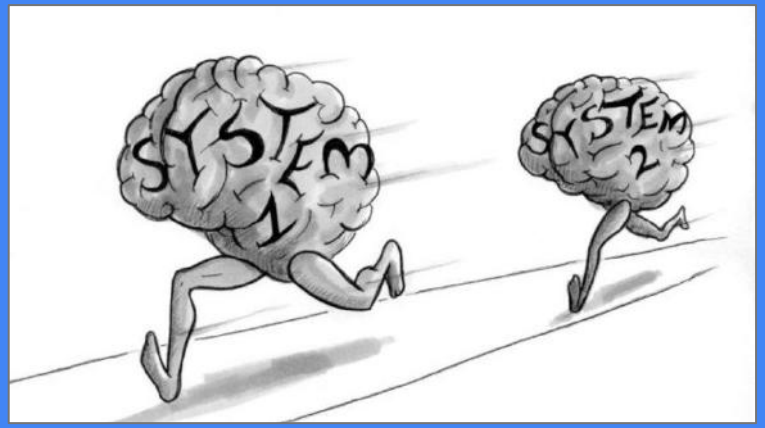
Daniel Kahneman, *Thinking, Fast and Slow*:

System 1

- Intuitive, fast, but has higher error rates

System 2

- Methodical, slow, low error rates, but is hard to engage



Code reviews make us more likely to generate “System 2 Code”

Also see:

http://bigthink.com/errors-we-live-by/kahnemans-mind-clarifying-biases

Rubber duck debugging

Explain the problem to a rubber duck:

- Go through problem step by step
- Think about each step clearly
- Realize what the problem is

This is **System 2** thinking!

Also see:

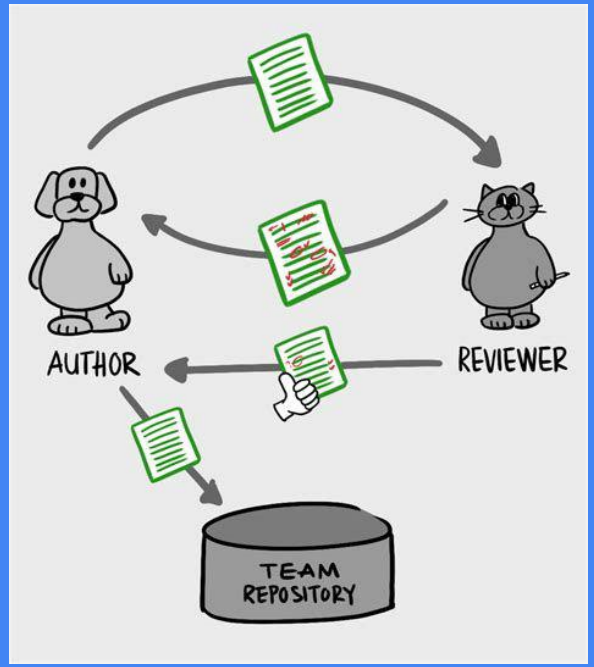
https://en.wikipedia.org/wiki/Rubber_duck_debugging

Research on code reviews

Code review catches 60-90% of errors.
Fagan 1976

The first reviewer and first review matter the most.
Cohen 2006

Defect rates in code are related to program size, and seemingly little else.
El Imam 2001



Also see:

http://www.mfagan.com/pdfs/ibmfagan.pdf

http://vimeo.com/9270320

http://www.slideshare.net/gvwilson/bits-of-evidence-2338367

Before sending code for review

- Write code that is easy to review!
- Keep changes small and focused
- Send a work in progress review out early
- Review your own work!

Mystery function

```
float mystery( float number )
{
     long i;
     float x2, y;
     const float threehalfs = 1.5F;

     x2 = number * 0.5F;
     y  = number;
     i  = * ( long * ) &y;          // evil floating point bit level hacking
     i  = 0x5f3759df - ( i >> 1 );               // what the ****?
     y  = * ( float * ) &i;
     y  = y * ( threehalfs - ( x2 * y * y ) );   // 1st iteration
//   y  = y * ( threehalfs - ( x2 * y * y ) );   // 2nd iteration (optional)

     return y;
 }
```

Change descriptions

- More than just “what” the change is…
- “Why” was the change made?
- “How” was the “why” accomplished?
- Any new testing?

… an inspired example:

```
#What Support @google.com login.

#Why Most users have google.com login and
user research shows preference for
passwordless login.

#How Give user options to enter password
or click to login with Google, initiating
oauth flow in <design doc link>.

#Testing

 - Added a test account
test-login@google.com

 - Added WebDriver tests that exercise the
login
```

During code review

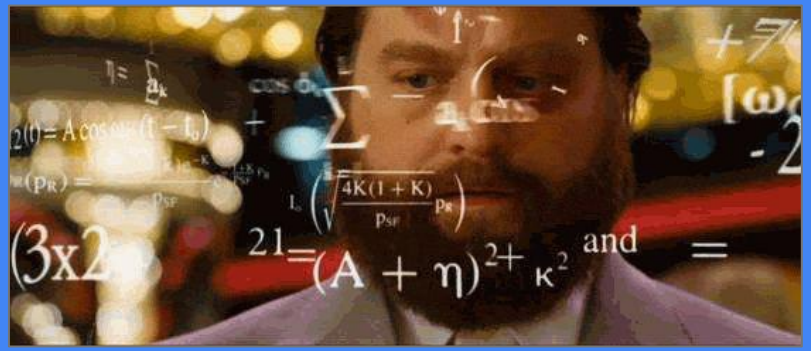
Flag errors of execution

- Unclear documentation
- Typos
- Style violations
- Bad/missing tests
- Bugs

During code review

Apply deliberative thinking to find errors

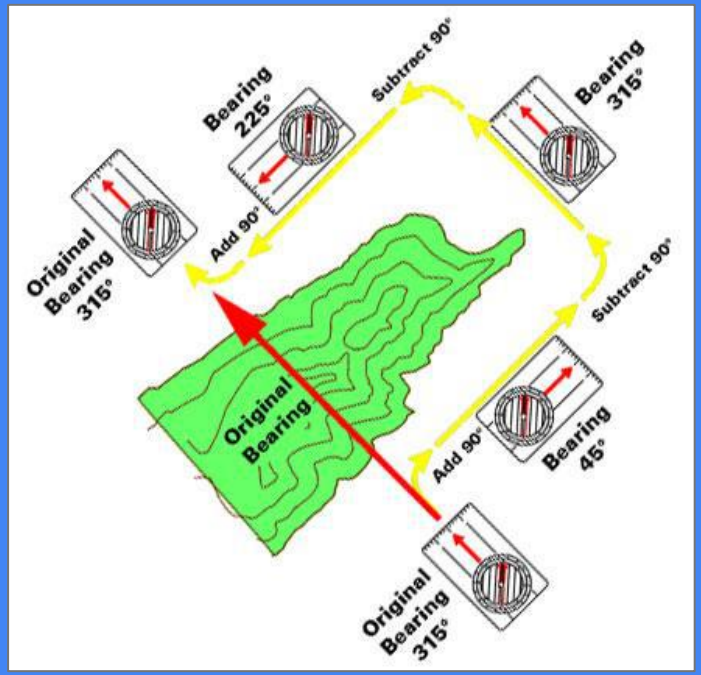
- Is this algorithm correct?
- Is this built to specifications?
- Does this code need to exist?
- Is this the most elegant solution?



During code review

Develop shared understanding about the purpose of the code

- Align team on “landmarks”
- Small changes can lead to target drift
- Each code review is an opportunity to course-correct
- How will this code be used next year?



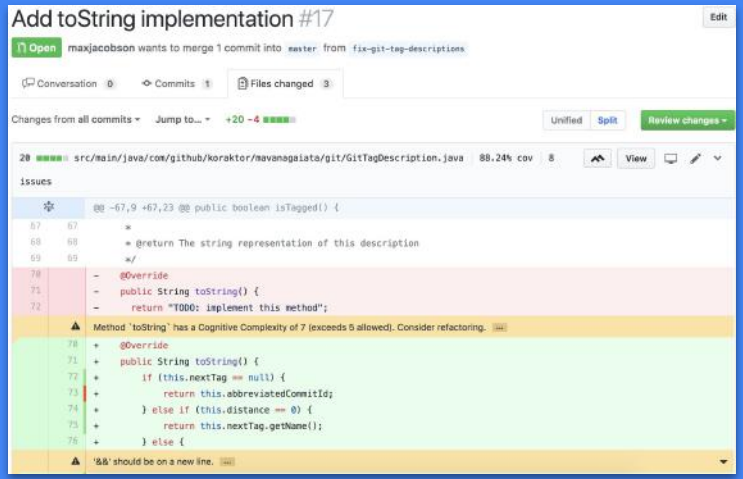
During code review

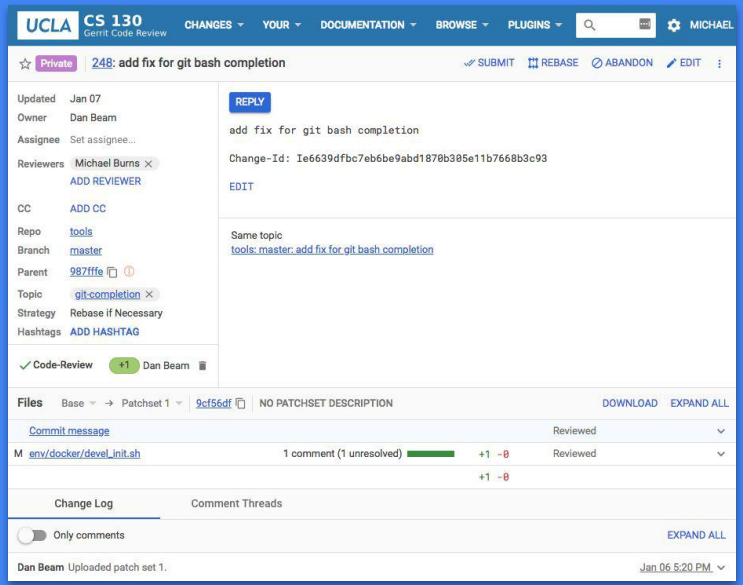
Establish N+1 availability on understanding of the code

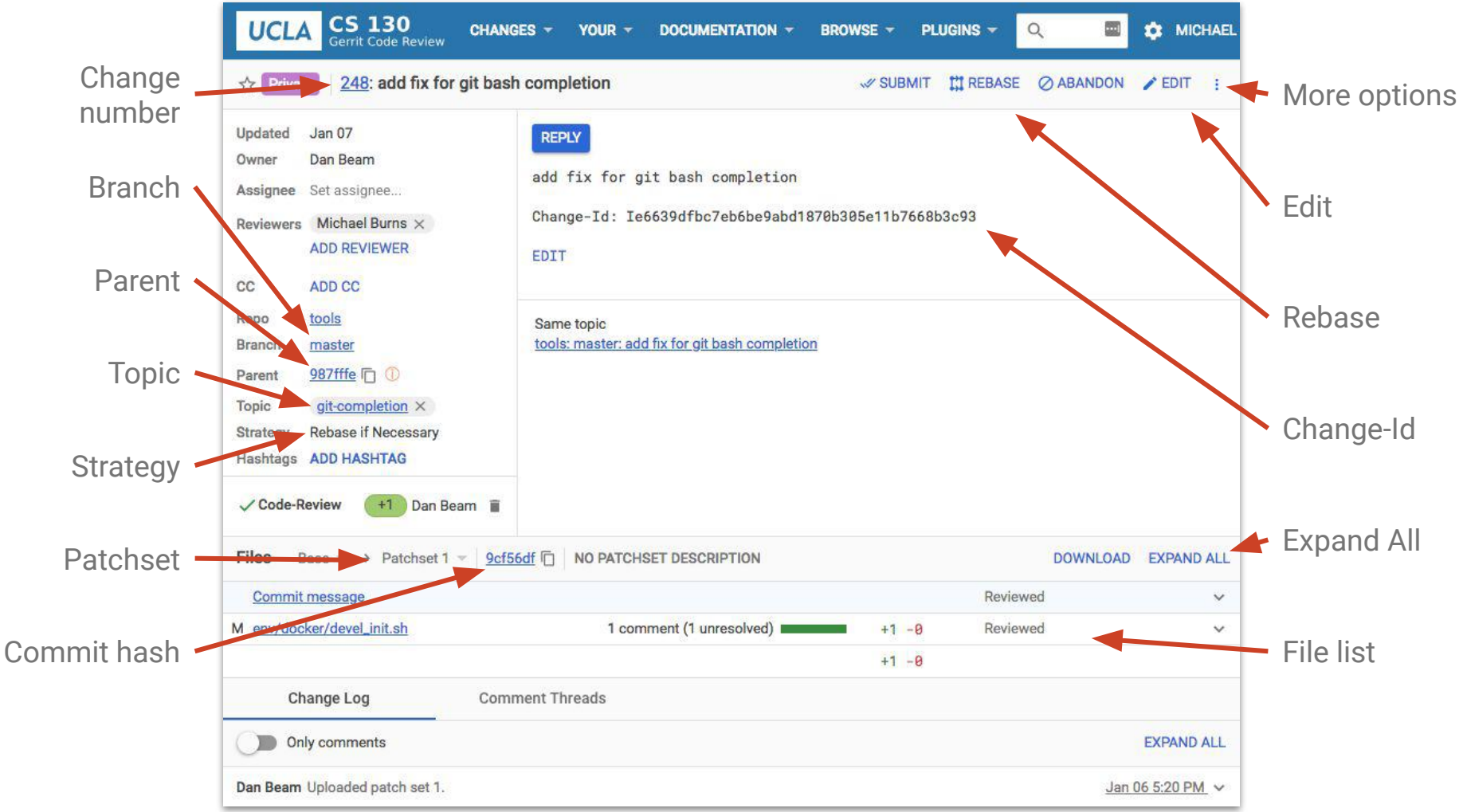
- Teams are dynamic
- Thanos number

Methods of code review

- Projecting code in a meeting
- Pair programming
- Pull requests
- Code review tools






UCLA
CS 130
Gerrit Code Review
CHANGES
YOUR
DOCUMENTATION
BROWSE
PLUGINS
MICHAEL
248: add fix for git bash completion
SUBMIT
REBASE
ABANDON
EDIT
Updated Jan 07
Owner Dan Beam
Assignee Set assignee...
Reviewers Michael Burns
ADD REVIEWER
CC ADD CC
Repo tools
Branch master
Parent 987fffe
Topic git-completion
Strategy Rebase if Necessary
Hashtags ADD HASHTAG
Code-Review +1 Dan Beam
REPLY
add fix for git bash completion
Change-Id: Ie6639dfbc7eb6be9abd1870b305e11b7668b3c93
EDIT
Same topic
tools: master: add fix for git bash completion
Files
Patchset 1
9cf56df
NO PATCHSET DESCRIPTION
DOWNLOAD
EXPAND ALL
Commit message
Reviewed
M env/docker/devel_init.sh
1 comment (1 unresolved)
+1 -0
Reviewed
+1 -0
Change Log
Comment Threads
Only comments
EXPAND ALL
Dan Beam Uploaded patch set 1.
Jan 06 5:20 PM
Change number
Branch
Parent
Topic
Strategy
Patchset
Commit hash
More options
Edit
Rebase
Change-Id
Expand All
File list

Pull Requests: Merge new code into the project

Final guidelines

- Be thoughtful and careful with words
- Avoid personal attacks
- Reviews are not a competition

But… do not be too easy either

Remember, you are not your code!

Break

Sample Code Reviews

Public code review

- Open source projects use code review
  - Chromium (Gerrit)
- We will also review student code

https://chromium-review.googlesource.com/c/chromium/src/+/1407963

https://chromium-review.googlesource.com/c/chromium/src/+/1406174

https://chromium-review.googlesource.com/c/chromium/src/+/1408032

Public code review

- Open source projects use code review
  - Chromium (Gerrit)
- We will also review student code

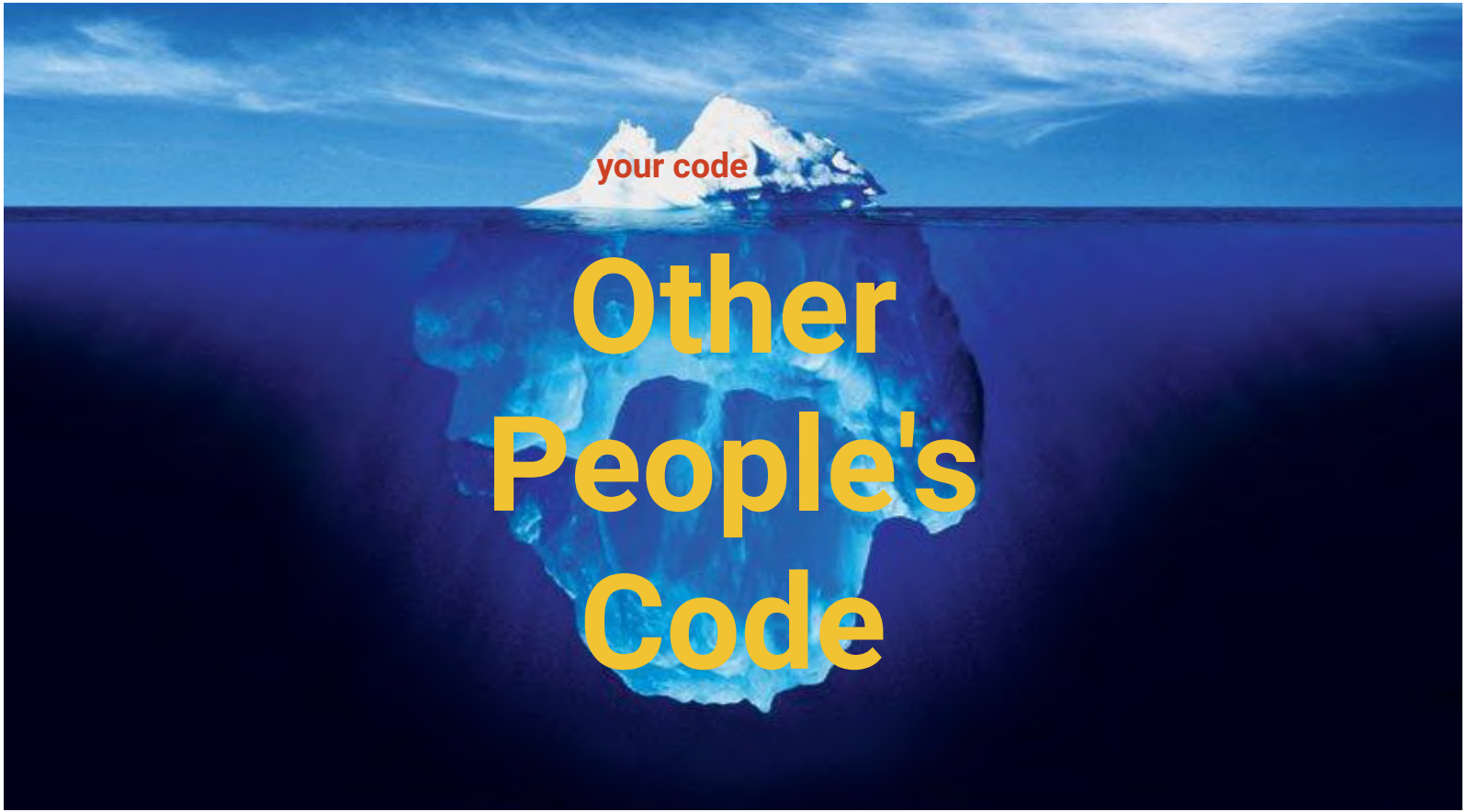
https://code.cs130.org/c/sb2714-config-parser/+/10352?tab=comments

https://code.cs130.org/c/yunqiu21-config-parser/+/10457

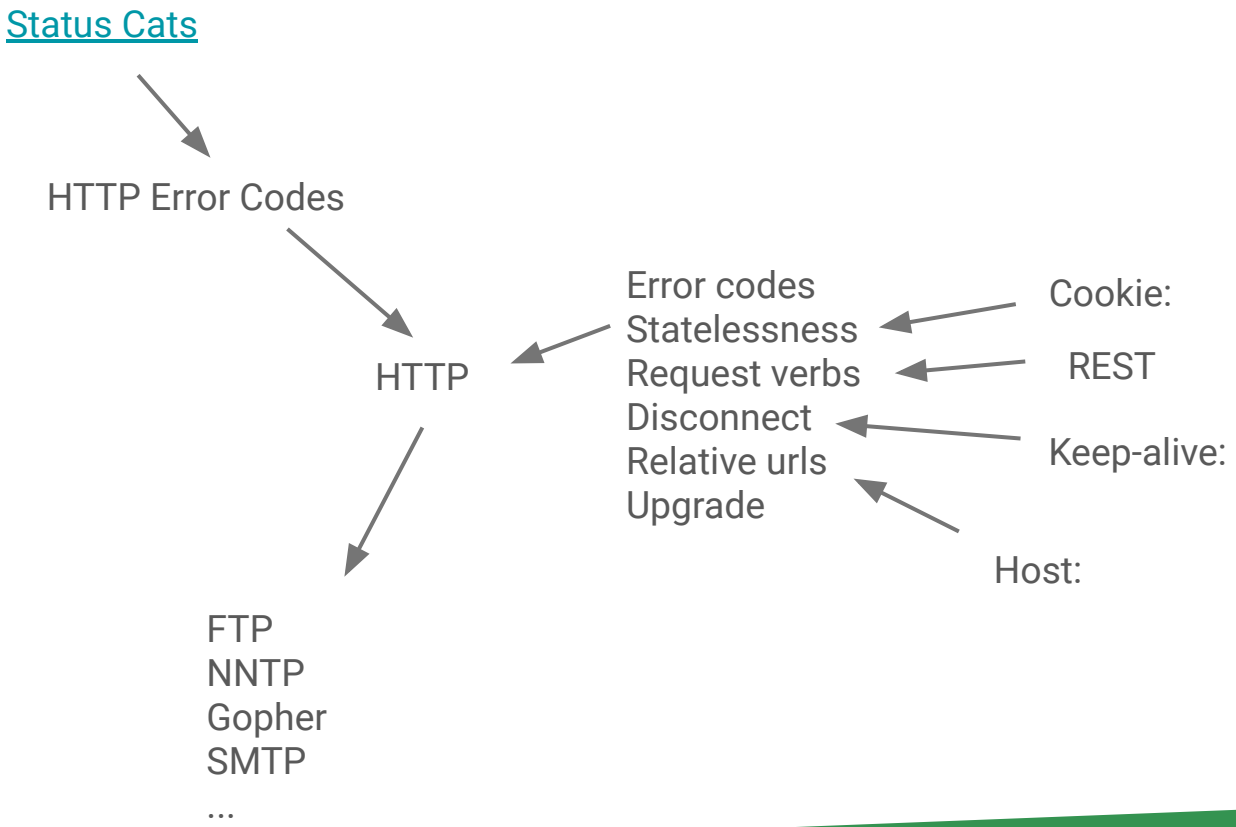
https://code.cs130.org/c/rohitjagan-config-parser/+/10535/2?tab=comments

https://code.cs130.org/c/timothypoon520-config-parser/+/10543

Webserver Development


your code
Other People's Code

HTTP in (some) context



Things you will see

- MIME types
- Header lines
- \r\n\r\n (or \n\n)
- Connection management
- Error codes
- Probers

Command-line HTTP tools

HTTP is human-readable!

Talk to your webserver personally

Craft responses to try things out

Watch log output as you issue queries

Put regressions into tests!

Netcat (nc)

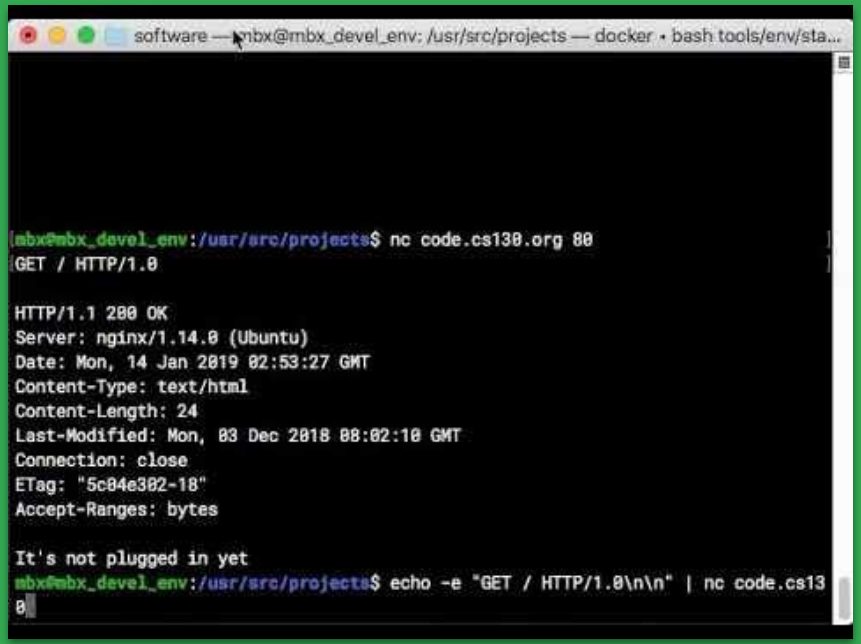
```
$ echo -e "GET / HTTP/1.0\n\n" | \
    nc code.cs130.org 80
```

Pros:

- Versatile
- Can easily craft malformed requests

Cons:

- Must manually craft well-formed requests
- No SSL support
- Knows nothing of HTTP



Curl (`curl`)

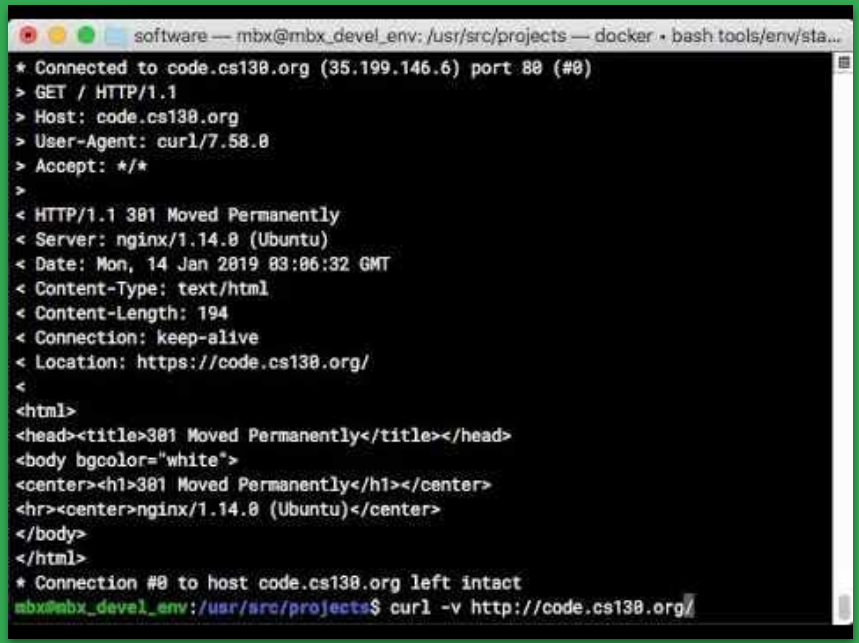
```
$ curl -v http://code.cs130.org/
```

Pros:

- Output can be concise
- Robust HTTP support
- Easy to use

Cons:

- Geared towards end-users, not developers
- Output can be hard to parse



HTTPie (`http`)

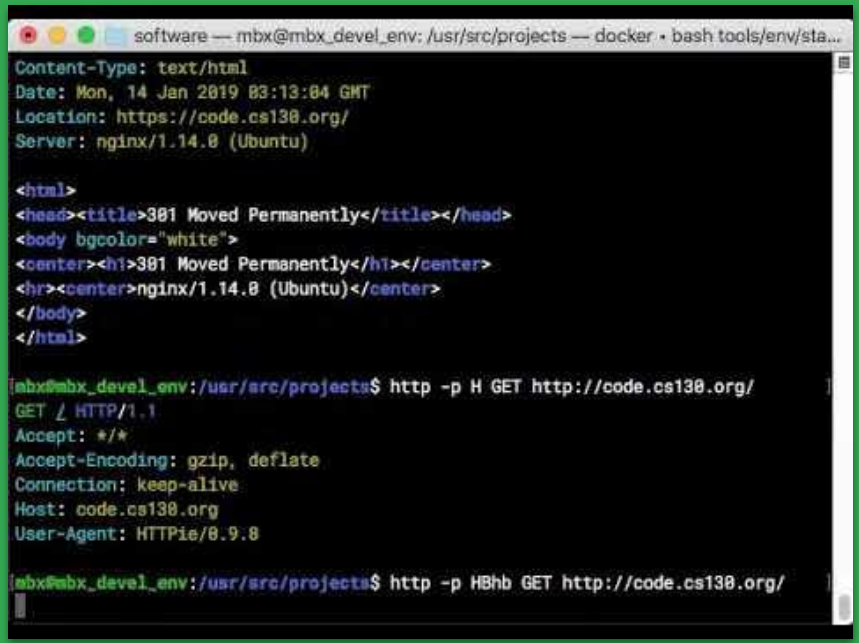
```
$ http -p HBhb GET \
    http://code.cs130.org/
```

Pros:

- Output is very configurable
- Robust HTTP support
- Can specify request method

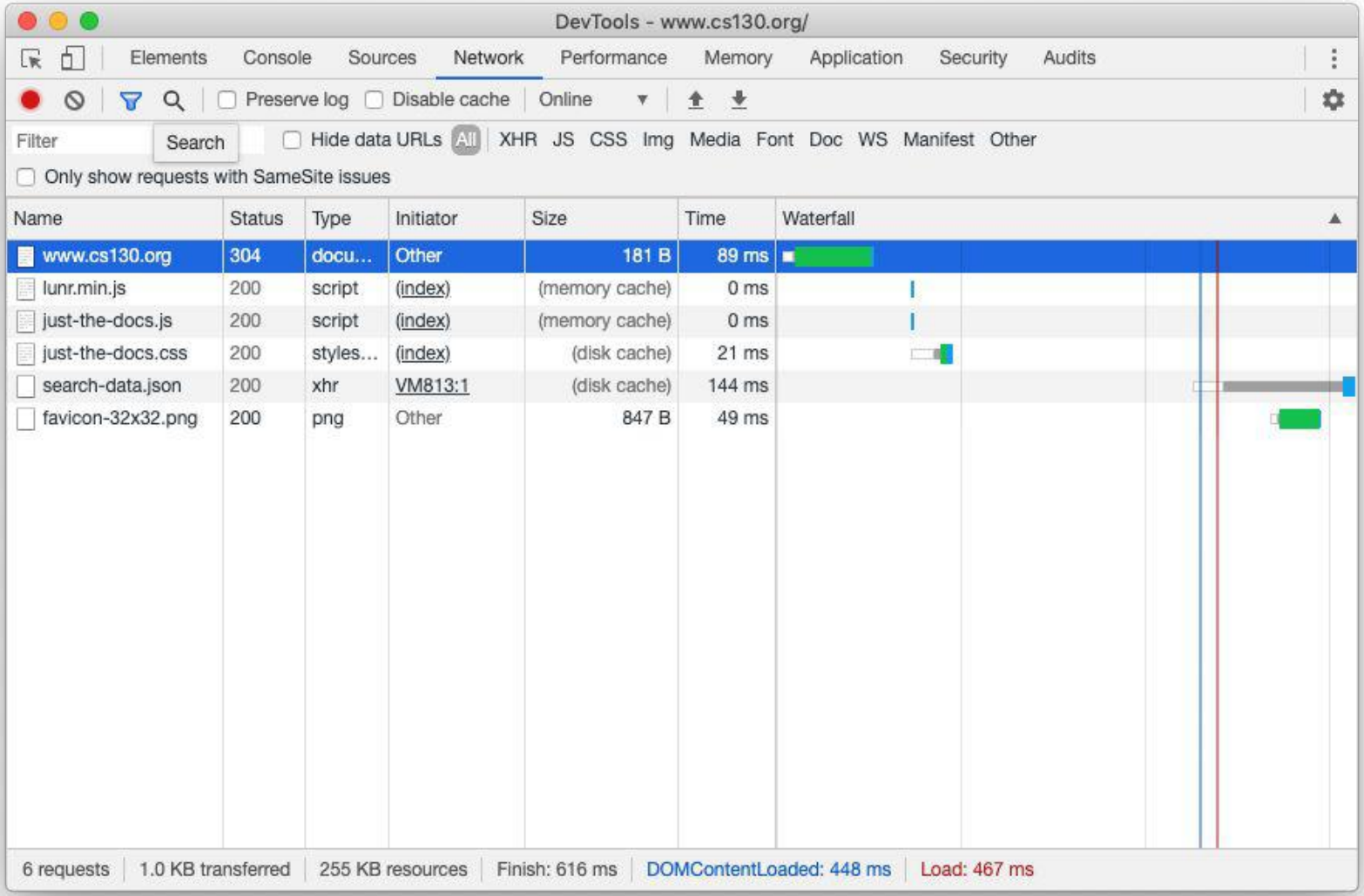
Cons:

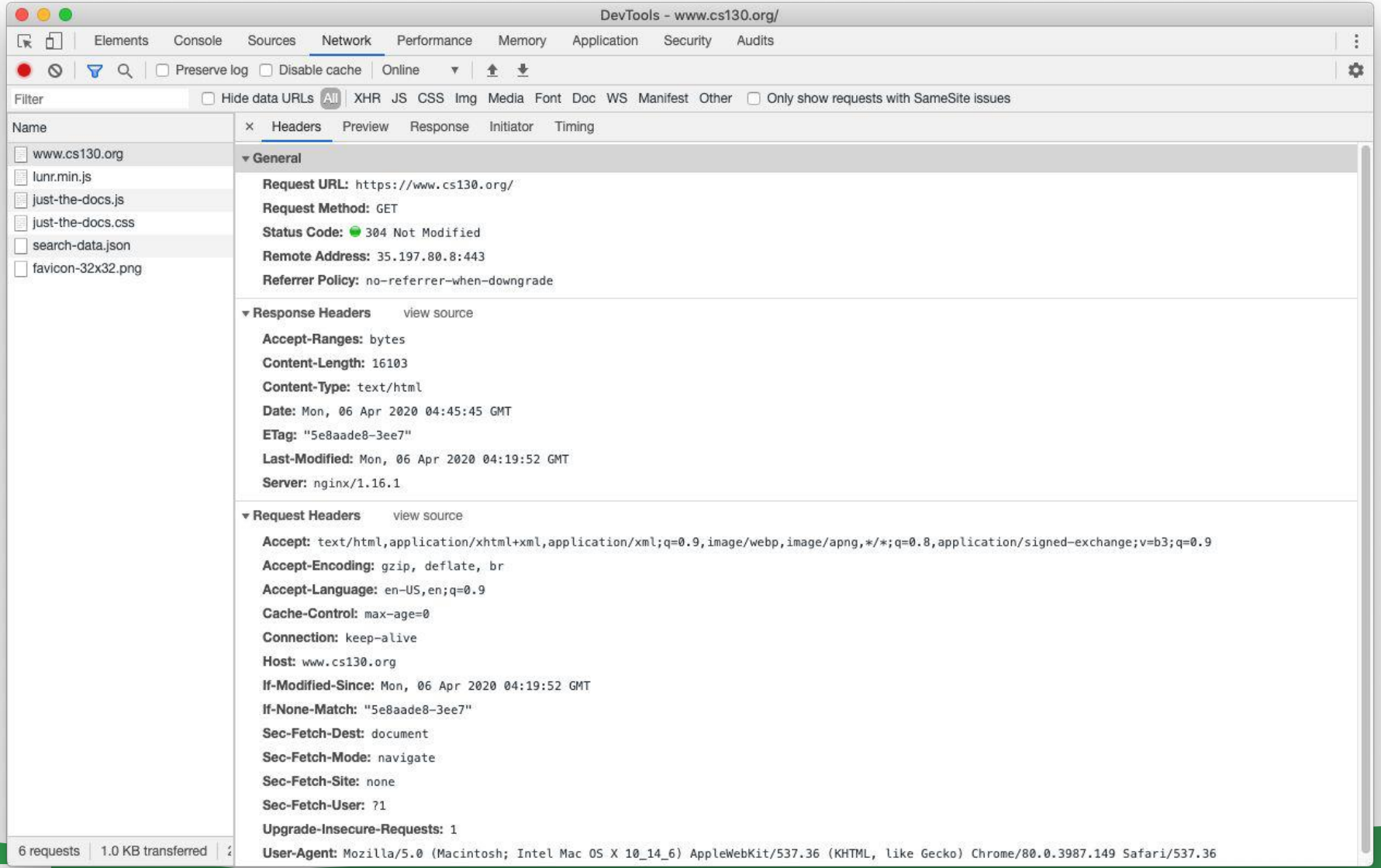
- Requires more configuration than other tools



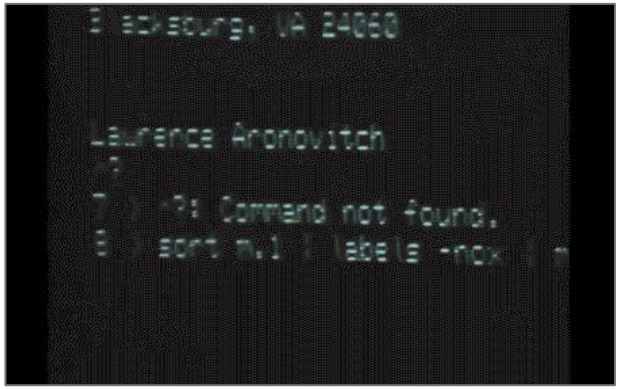
Using real browsers

- Early goal should be to support browser requests
- Tough on web servers
- Implement many (all?) aspects of HTTP


DevTools - www.cs130.org/
Elements
Console
Sources
Network
Performance
Memory
Application
Security
Audits
Preserve log
Disable cache
Online
Filter
Search
Hide data URLs
All
XHR
JS
CSS
Img
Media
Font
Doc
WS
Manifest
Other
Only show requests with SameSite issues
Name
Status
Type
Initiator
Size
Time
Waterfall
www.cs130.org 304 docu... Other 181 B 89 ms
lunr.min.js 200 script (index) (memory cache) 0 ms
just-the-docs.js 200 script (index) (memory cache) 0 ms
just-the-docs.css 200 styles... (index) (disk cache) 21 ms
search-data.json 200 xhr VM813:1 (disk cache) 144 ms
favicon-32x32.png 200 png Other 847 B 49 ms
6 requests
1.0 KB transferred
255 KB resources
Finish: 616 ms
DOMContentLoaded: 448 ms
Load: 467 ms


DevTools - www.cs130.org/
Elements
Console
Sources
Network
Performance
Memory
Application
Security
Audits
Preserve log
Disable cache
Online
Filter
Hide data URLs
All
XHR
JS
CSS
Img
Media
Font
Doc
WS
Manifest
Other
Only show requests with SameSite issues
Name
www.cs130.org
lunr.min.js
just-the-docs.js
just-the-docs.css
search-data.json
favicon-32x32.png
Headers
Preview
Response
Initiator
Timing
General
Request URL: https://www.cs130.org/
Request Method: GET
Status Code: 304 Not Modified
Remote Address: 35.197.80.8:443
Referrer Policy: no-referrer-when-downgrade
Response Headers
view source
Accept-Ranges: bytes
Content-Length: 16103
Content-Type: text/html
Date: Mon, 06 Apr 2020 04:45:45 GMT
ETag: "5e8aade8-3ee7"
Last-Modified: Mon, 06 Apr 2020 04:19:52 GMT
Server: nginx/1.16.1
Request Headers
view source
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,image/apng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.9
Accept-Encoding: gzip, deflate, br
Accept-Language: en-US,en;q=0.9
Cache-Control: max-age=0
Connection: keep-alive
Host: www.cs130.org
If-Modified-Since: Mon, 06 Apr 2020 04:19:52 GMT
If-None-Match: "5e8aade8-3ee7"
Sec-Fetch-Dest: document
Sec-Fetch-Mode: navigate
Sec-Fetch-Site: none
Sec-Fetch-User: ?1
Upgrade-Insecure-Requests: 1
User-Agent: Mozilla/5.0 (Macintosh; Intel Mac OS X 10_14_6) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/80.0.3987.149 Safari/537.36
6 requests
1.0 KB transferred

Sensible program



Your webserver

Write Logs!

- “printf debugging”
  - What is going wrong
  - What is going right!
- Unmasks “asserts” in your code -- finds problems before you do
- Lets you keep track of events in a headless program
- Lots of good libraries for log granularity, formatting, etc.
- Never gets old ...

Coming up

Assignment 2

Assigned tomorrow, due next Tuesday

https://bit.ly/38gNRn5

A tweet: Ask the rubber duck a question.

A word: Name a useful testing or debugging tool (for software or for life)