

# Text-Based Web Browser

Team Good Meowning

By Yichen Cui, Andi Fan, Nicki Nguyen, Myles Thiessen



# What did we build?

```

Lynx (web browser) - Wikipedia, the free encyclopedia (p1 of 5)

#copyright

Your continued donations keep Wikipedia running!

Lynx (web browser)

From Wikipedia, the free encyclopedia

Jump to: navigation, search

CAPTION: Lynx

Wikipedia Main Page displayed in Lynx
Wikipedia Main Page displayed in Lynx
Maintainer: Thomas Dickey
Stable release: 2.8.5 (February 4, 2004) [[+/-]]
Preview release: 2.8.6 (?) [[+/-]]
OS: Cross-platform
Use: web browser
License: GPL
Website: lynx.isc.org

Lynx is a text-only Web browser and Internet Gopher client for use on cursor-addressable, character cell terminals.

Browsing in Lynx consists of highlighting the chosen link using cursor keys, or having all links on a page numbered and entering the chosen link's number. Current versions support SSL and many HTML features. Tables are linearized (scrunched together one cell after another without tabular structure), while frames are identified by name and can be explored as if they were separate pages.

Lynx is a product of the Distributed Computing Group within Academic Computing Services of the University of Kansas, and was initially developed in 1992 by a team of students at the university (Lou Montulli, Michael Grobe and Charles Rezac) as a hypertext browser used solely to distribute campus information as part of a Campus-Wide Information Server. In 1993 Montulli added an Internet interface and released a new version (2.0) of the browser [1] [2] [3].

more- http://en.wikipedia.org/wiki/Image:Lynx_Z28web_browser_Z29.png
```

Lynx Web Browser  
<https://lynx.invisible-island.net/>

```

Tab 1/1 - http://www.cs.toronto.edu/~arnold/
Arnold Rosenbloom

CS Educators

EC00 2012 Recursion Notes[0]{"liason/ec002012/"

Please let me know ( mail_arnold@cs.toronto.edu[1]{"mailto:arnold@cs.toronto.edu"} ) if
there is interest in an advanced CS session for CIT this summer. If so, UTM will hold only
the advanced session, the introductory session will be on StGeorge.

Notes from BEIT 2003[2]{"http://www.cs.toronto.edu/~arnold/teaching/pss/beit03"}

Computing Insights For Teachers (CIT)[3]{"cit/"

CS Students: Things of interest

Some thoughts/sayings of mine: 1) Do something, something happens. Do nothing, nothing
happens. 2) Wisdom is where you find it. Happiness is where you find it. 3) If you have not
failed three times before lunch, you are not trying hard enough. 4) Having a brain is not
easy. Having two is next to impossible. 5) Indifference is the true mark of confidence. 6)
Never miss an opportunity to do the right thing. 7) To become an expert, repeat until it is
easy. 8) An expert knows how to apply minimal effort for maximum effect.
```

Press [h] for help menu

Good Meowning's  
Text-based Web Browser

# Why does our project matter?

- To take a deep dive into learning about and implementing the web browser
  - Understand the basic concept of web browser: fetching data through HTTP requests, processing HTML data, displaying data in the terminal
  - Revealed the difficulty behind implementation of some core features present in modern browsers
- Text-based web browsers are still in-use today! For instance, the Gentoo Linux installation handbook suggest users to use a terminal web browser (Lynx) for initial tasks like downloading stage files for installation

# What are our restrictions?

## 1. HTML only!

- No CSS, JavaScript, JSON, etc...
- No JSON
- No cookies
- No local storage

## 2. No media

- No images, animations, videos, etc...



**DEMO**

# Major Features

- Visit web pages through our terminal browser
  - Accept an URL to a website or a path to a local file
  - Display the text from the HTML file at the specified URL or file path
  - Display the current URL visited on the top left of the screen
  - Include an insert-mode to enter URLs or embedded hyperlinks
- Interact with web pages through our terminal browser
  - Cycle through and visit embedded hyperlinks
  - Scroll up/down the screen
- Quality-of-life features
  - Include a help box to display commands (stretch goal)
  - Multiple browsing tabs (stretch goal)
  - Keep a list of browsing history (stretch goal)

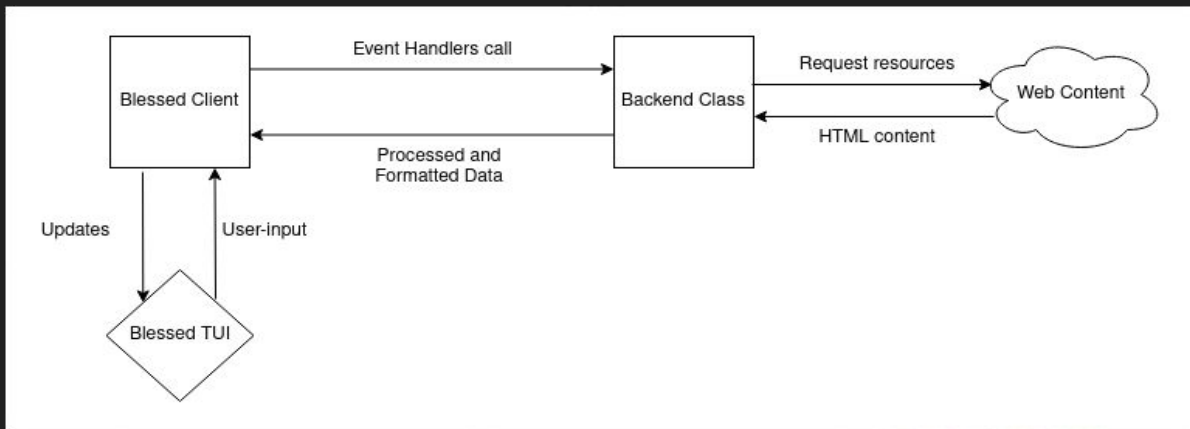
# Major Components

## 1. Frontend User Client

- Using Blessed as our TUI library
- Initializes blessed components and event handlers
- Displays parsed web content

## 2. Backend Class

- Retrieves HTML content, and stores in a hierarchical structure using Cheerios
- Sanitizes user input
- Standardizes respond before passing to blessed-client
- Parses content tree and assigns blessed styling tags to HTML elements



```
src
├── app.ts
├── backend-class
│   ├── blessed-server.ts
│   ├── data-fetcher.ts
│   ├── data-parser.ts
│   └── index.ts
├── blessed-client
│   ├── blessed-client.ts
│   ├── box-attributes.ts
│   └── index.ts
├── test
│   ├── cases-data-parser
│   │   ├── cases-axios.ts
│   │   ├── cases-fs.ts
│   │   ├── cases-href.ts
│   │   └── cases-url.ts
│   ├── cases-data-server
│   │   ├── cases-axios.ts
│   │   └── cases-fs.ts
│   ├── test.ts
│   └── test.sh
```

# Adjustments

## 1. Remove HTML forms as a stretch goal feature

- HTML forms without JavaScript are not useful
- Difficult to implement due to variations

A form with input fields for text:

```
<form>
  <label for="fname">First name:</label><br>
  <input type="text" id="fname" name="fname"><br>
  <label for="lname">Last name:</label><br>
  <input type="text" id="lname" name="lname">
</form>
```



# Adjustments

## 2. Interact with hyperlinks through key-presses instead of mouse clicks

- Ideally we wanted the user to be able to click on the hyperlinks, but there is no support for this feature in the frontend library
- Workaround ideas:
  - Use a Layout Element as a container class, involves nesting Blessed elements
  - Calculating the index of the hyperlink using the 2D cursor click position
  - Hashing text around hyperlinks to use text snapshots around click position to search for the corresponding hyperlink

```
Tab 1/1 - https://q.utoronto.ca/courses/239112/assignments/742388
UTORid Account Recovery Service
If you have registered with UTORid Account Recovery Service[7]("https://www.utorid.utoronto.ca/accountrecovery"), you can use the UTORid password reset tool[8]("https://recover.utorid.utoronto.ca/").
If you are not registered, ...
If you are not registered with the UTORid Account Recovery Service[9]("https://www.utorid.utoronto.ca/accountrecovery"), please visit your campus help desk:
St. George[10]("http://help.ic.utoronto.ca/") UTM[11]("http://www.utm.utoronto.ca/ilts/") UTSC[12]("http://www.utsc.utoronto.ca/ilts/")
Press [h] for help menu
```

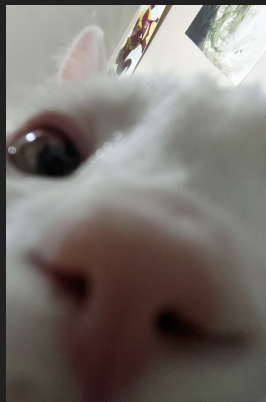
l[8]("https://r

# Testing and Validation

- Verified acceptance criteria for features with industry partner
- Branch protection rules
- Manual QA
- Units tests written for backend features
  - Covers all edge-cases that we can think of, and all past bugs we have experienced
  - Overall branch code coverage of 88% according to istanbul.js

```
// Remote files via Axios
describe("Test remote files", () => {
  // Positive cases
  casesDataServerHTTP.forEach(
    ([url, parsedURL, output, outputForward, outputBackward, href]) => {
      it(url, async () => {
        const dataServer = new DataServer();
        await dataServer.visitURL(url);
        chai.assert.deepEqual(dataServer.renderPage(), [output, parsedURL]);
        chai.assert.deepEqual(dataServer.renderPage(6), [
          outputForward,
          parsedURL
        ]);
        chai.assert.deepEqual(dataServer.renderPage(-8), [
          outputBackward,
          parsedURL
        ]);
        chai.assert.equal(dataServer.getHrefURL(), href);
      });
    }
  );
});
```





Thank you for listening



Special thanks to Mike, Greg, and Julia for your help and guidance!

