

CS1101S Discussion Group Week 13: *Web Development & Final Review*

Niu Yunpeng

niuyunpeng@u.nus.edu

November 14, 2017

- 1 Web development
 - Overview
 - Basic knowledge - static page
 - Advance knowledge - dynamic page
- 2 Information security
 - Web security
- 3 Final review
 - What we have learned
 - To prepare for final examination

What is the trend of this industry?

- Desktop first?
- Web first?
- Mobile first?
- ...



Answer

- Web first? Yes.
- Mobile first? Yes.
- ...

Result...

- The web goes mobile.

How does the Internet work?

- Magic.
- A lot of magic.

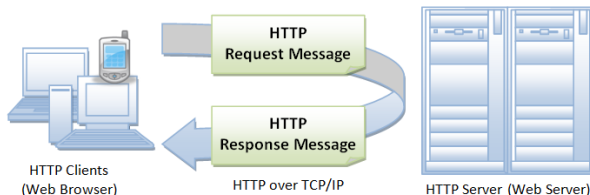


Before we go on...

- URL: Uniform Resource Locator
- HTTP: HyperText Transfer Protocol
- DNS: Domain Name System
- HTML: HyperText Markup Language
- CSS: Cascading Style Sheets
- ...

Basic idea

- You enter an URL: request for something by its unique identifier.
- Go through DNS and reach the server.
- Find the requested resources on the server and return.
- Render the resources in the browser.



From the most basic - static webpage

You have three “weapons” with you as follows:

- HTML(5) for content
- CSS(3) for style
- JavaScript for action



→ They make our site more

Your first webpage

```
<!DOCTYPE html>
<html>
<head>
  <title>My heading</title>
</head>
<body>
  <p>Hello , world!</p>
</body>
</html>
```

Instructions

- Save as *<something>.html* locally.
- Open it in the browser.

Make it prettier - apply style on elements

Edit and save as `<something>.css` locally:

```
p {  
  color: #00ffff  
}
```

Use the CSS style defined just now

In `<something>.html`, modify it to become:

```
<head>  
  <title>My heading</title>  
  <link rel="stylesheet" type="text/css"  
        href="something.css">  
</head>
```

Embed CSS style in HTML

Edit and save as `<something>.html` locally:

```
<!DOCTYPE html>
<html>
<head>
  <title>My heading</title>
</head>
<body>
  <p style="color: red;">Hello, world!</p>
</body>
</html>
```

Embed JavaScript in HTML

Edit and save as `<something>.html` locally:

```
<!DOCTYPE html>
<html>
<head>
  <title>My heading</title>
</head>
<body>
  <p>Hello, world!</p>
  <button onclick="my_alert();">Click here</button>
  <script type="text/javascript">
    function my_alert() {
      alert("Haha. This is interesting!");
    }
  </script>
</body>
</html>
```

Dynamic webpage

- Our website should display customize content for different users.
- The return resources may vary even though you enter the same URL.

Solution

- Process the files before giving response to the users.
- Use server-side programming.

Thanks to dynamic webpage...

- The world of Internet has become so colorful.



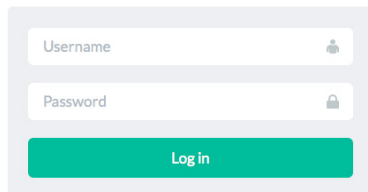
To make a dynamic webpage...

You need to learn the things below:

- Select a server-side programming language: *PHP, Ruby, Java, ...*
- Select a database engine: *MySQL, PostgreSQL, MariaDB, ...*
- Select a server: *Apache, Tomcat, Nginx, ...*

How to distinguish different users

- Prompt users to sign in when they visit the website.
- Save some informaton in COOKIE or SESSION.
- Whenever one user enters a URL, display the correct content based on the information stored in COOKIE or SESSION.
- Clear the COOKIE or SESSION when the user tries to logout.



A login form consisting of two input fields and a button. The first field is labeled 'Username' and has a small person icon to its right. The second field is labeled 'Password' and has a small lock icon to its right. Below these fields is a green button with the text 'Log in' in white.

Recommended modules at SoC

- CS3226 Web Programming and Applications

Prestigious opportunity at SoC

- Computing for Voluntary Welfare Organisations (CVWO)

What to expect at CVWO

- Build web-based administration system for VWOs (Voluntary Welfare Organisations) at Singapore.
- Get awesome hand-on experiences even from Year 1.
- Give back to the society using your CS knowledge.
- Receive a stipend of \$????/month for 3 months.
- Claim 6MCs as SIP (Student Internship Program CP3200).
- Claim CIP hours (for foreign scholars).

To join CVWO in Year 1 summer

- Obtain **really very good** grades in CS1101S, CS2030, CS2040.
- Do two or three assignments during December break.
- Apply in March and wait for interview in April.
- Show strong programming skills and adequate project experiences.



Last week - CVWO 10th Anniversary



- 1 Web development
 - Overview
 - Basic knowledge - static page
 - Advance knowledge - dynamic page
- 2 Information security
 - Web security
- 3 Final review
 - What we have learned
 - To prepare for final examination

Web security

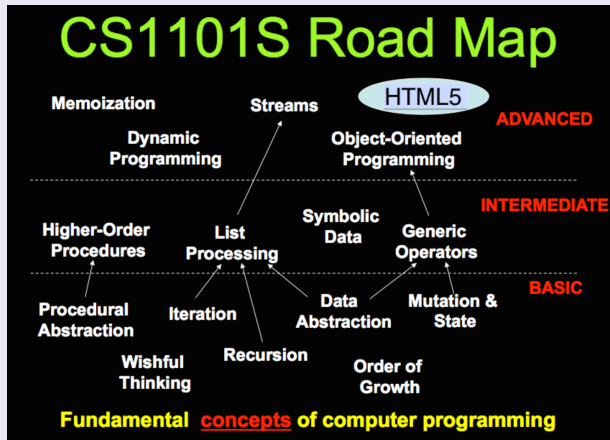


An interesting problem

- Click here http://cs1101s.azurewebsites.net/web_sec/

- 1 Web development
 - Overview
 - Basic knowledge - static page
 - Advance knowledge - dynamic page
- 2 Information security
 - Web security
- 3 Final review
 - What we have learned
 - To prepare for final examination

Revisit the CS1101S roadmap



Things we have covered in this semester

- Components of programming language
- Wishful thinking/abstraction
- Recursion/iteration
- Higher-order programming
- Pair/list/tree processing
- Data structure design
- Memoization/dynamic programming
- Object-oriented programming
- Array % loop
- Meta-Circular Evaluator
- ...

Components of programming language

- Primitives:
The smallest constituent unit of a programming language.
- Combination:
Ways to put primitives together.
- Abstraction:
The method to simplify the messy combinations.
 - To abstract data: use naming;
 - To abstract procedures: use functions.
 - Sometimes, naming and functions are combined together.

Wishful thinking/abstraction

To make a good abstraction:

- Modularity:
Separate multiple steps (and sub-steps).
- Readability:
Easy for others to read and understand.
- Reusability:
Provide a generic interface to be used commonly.
- Maintainability:
Convenient to debug, refactor and deploy.

Recursion

- Due to the top-down approach.
- Use substitution model to understand.
- Use pair/list/tree/stream processing.
- May give rise to recursive or iterative call (due to tail recursion).

Iteration

- Due to the bottom-up approach.
- Use environment model to understand.
- Use array and loop.

Classical examples of recursion

Can they be solved using iteration as well?

- Factorial
- Square root
- Power function
- Fibonacci
- Greatest common divisor (GCD)
- Least common multiple (LCM)
- Hanoi tower
- Coin change
- Permutation / combination

Higher-order programming

Why we can do higher-order programming:

- Functions are also variables.
- They are not special.
- They just behave like normal variables.

To use higher-order programming:

- Variables can be functions.
- Parameters can be functions.
- Return values can be functions.

Pair/list/tree processing

Up to now, the list library supports different kinds of functions:

- List builder: `list`, `build_list`, `enum_list`;
- List getter: `head`, `tail`, `list_ref`, `member`, `is_member`;
- List information: `is_list`, `is_empty_list`, `length`;
- List modifier: `append`, `reverse`, `remove`, `remove_all`, `filter`, `map`, `for_each`;
- List converter: `accumulate`, `list_to_string`.

Stream processing

Up to now, the stream library supports different kinds of functions:

- Stream builder: `stream`, `build_stream`, `enum_stream`, `integers_from`;
- Stream getter: `stream_tail`, `stream_ref`, `stream_member`;
- List information: `is_stream`, `stream_length`;
- List modifier: `stream_append`, `stream_reverse`, `stream_map`, `stream_for_each`, `stream_remove`, `stream_remove_all`, `stream_filter`;
- List converter: `list_to_stream`, `stream_to_list`, `eval_stream`.

Data structure design

You should follow these principles:

- Understand the requirement before doing the actual design;
- Separate the interface from the implementation;
- Compare the advantage and tradeoff;
- Principle of last commitment.

Memoization/dynamic programming

Use a (1D/2D) table to save all previously computed results:

```
function memoize(func) {
  var table = make_table();

  return function (x) {
    if (contains(x, table)) {
      return lookup(x, table);
    } else {
      var result = func(x);
      put(x, result, table);

      return result;
    }
  };
}
```

Object-oriented programming

- The constructor should be the same name as the class.
- Use `new` and call the constructor to instantiate an object.
- To inherit from the superclass, call the superclass's constructor and then use the *Inherits* method to link the prototype chain.
- To implement polymorphism, override the method in the subclass.
- Use `<class_name>.prototype.<function_name>.call` to call the version from the ancestor.

Meta-Circular Evaluator

- Add tags to differentiate various statements.
- Primitive data types/operators are self-evaluating.
- Use list and table to implement environment model.

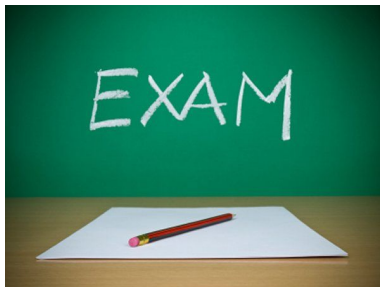
- Basic evaluator?
- OOP evaluator?
- Lazy evaluator?
- Memoized evaluator?
- Tail recursion evaluator?

To prepare for the final examination

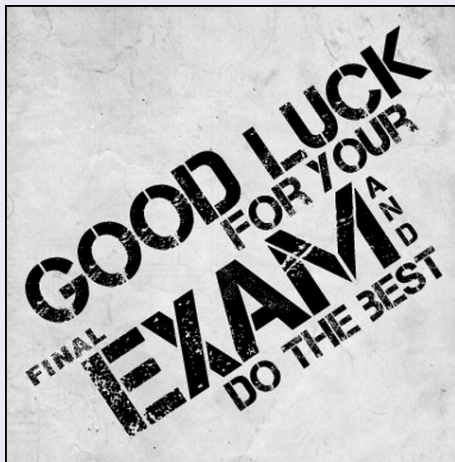
- Read all the materials distributed again;
- Do as many PYPs (past year papers) as possible;
- Summarize what you have learned;
- Be relaxed.

One weird thing about CS1101S final examination

- For most modules, the final will be much harder than the mid-term.
- However, CS1101S final may not be that difficult.
- ...



May the Source be with you!



No DG Problem for Week 13!

End

The End