Information Retrieval

WS 2016 / 2017

Lecture 6, Tuesday November 29th, 2016

(How to build a web application)

Prof. Dr. Hannah Bast
Chair of Algorithms and Data Structures
Department of Computer Science
University of Freiburg

UNI FREIBURG

Overview of this lecture

UNI FREIBURG

- Organizational
  - Your experiences with ES5 fuzzy prefix search
- Contents
  - How to build a search web application

    Sockets create, accept, receive, send, close

    Hypertext HTTP, Mime types, HTML, CSS

    JavaScript DOM, AJAX, JSON, jQuery
  - ES6: build a web app that displays fuzzy prefix matches (ES5) as you type your query

    This will be exciting but quite a bit of work, hence we split it over ES6 and ES7, see the explanations on ES6

Experiences with ES5 1/3

- Summary / excerpts
  - Interesting exercise with a cool result
  - Useful for (deeper) understanding of concepts from L5
  - Quite time-intensive for many ... usually worth it
  - Detailed explanations and TIP very file helpful for most
  - Not sure which deviations from TIP file are allowed

    We tried to make this clearer on the TIP file for ES6

    There will also be an award this time if you find errors
  - Why would I walk or wear shoes if I were a motor protein?
  - Biology student X: DNA slides in L5 were "borderline wrong"

    Detailed answer to ES5 question: https://goo.gl/opwE27

Experiences with ES5 2/3

- Results
  - Percentage of names, for which PED was computed

london england	9.2%	(ambiguous query)
freib	1.2%	(typical query)
rotenburg ober tauba	0.5%	(typical query)

  - Query times < 100 ms, even for london england query
  - Java vs. C++: a lot of variation in your running times

    Some Java codes are faster than some of the C++ codes

    The average C++ code is faster than the average Java code

    The fastest Java code is as fast as the fastest C++ code

Experiences with ES5 3/3

- Motor protein
  - Each cell has a cytoskeleton = a 3D network of "roads"
  - Two types of roads: microfilaments ("threads" of ~ 6 nm diameter) and microtubules ("cylinders" of ~ 24 nm diam.)
  - Two kinds of motor proteins: Myosin "walks" along micro-filaments, Dynein/Kinesin "walk" along microtubules
  - They look like stick figures, with a shoe size of 2 - 6 nm and a step size of 10 - 40 nm
  - https://www.youtube.com/watch?v=FzcTgrxMzZk&t=2m20s
  - https://www.youtube.com/watch?v=tMKlPDBRJ1E&t=1m15s

This is one of the central mechanisms of (your) life

Search web application

- Main components
  - Server that delivers the web pages
  - The contents of the web pages
  - The code that runs as part of the web pages and communicates with the server that answers queries
- Implementation
  - Many technologies behind this, each quite complex
  - But the basic principle behind each is easy to understand

  In the following, brief motivation + example for each

  Along with that we will code a toy web application **live**

Socket Communication 1/5

- Motivation
  - Two programs / processes communicating with each other, possibly (and often) on two different machines
  - For a typical web application:

    Browser asking for (static) web pages

    Code in web page asking for (dynamic) contents
  - Endpoint of such a communication channel is called **socket**
  - Each socket belongs to a particular machine (host) and has a unique id (port) on that machine

    The same machine can have many communication channels, hence the concept of (many) ports

Socket Communication 2/5

- High-level procedure
  - **Server side:**

    Create a socket and bind it to a give port

    Listen on that port for incoming requests

    Read request, compute result, send result
  - **Client side:**

    Connect to socket on server (need machine name + port)

    OS automatically assigns unique port on client machine

    Send request, wait for result

Socket Communication 3/5

- Implementation, server side
  - All programming languages have standard libraries for convenient socket communication (for server and client)

Python	socket	
Java	java.net.ServerSocket	
C++	boost::asio	(asio = asynchronous IO)

We provide code for the server socket communication on the Wiki, in both Java and C++

Since for ES6 you have to integrate your solution from ES5, Python is not a meaningful option for this ES6

Let's now live-code a simple server in Java ...

Socket Communication 4/5

- Implementation, server side, Java
  - Create socket, wait for request, get request, send result

```
ServerSocket  server = new ServerSocket(port);

Socket client = server.accept();

BufferedReader input = new BufferedReader(
   new InputStreamReader(client.getInputStream()));
DataOutputStream output = new DataOutputStream(
                                client.getOutputStream());

String request = input.readLine();          // Read string.

output.write("...".getBytes("UTF-8"));      // Write string.
output.write(... response bytes ...);       // Write bytes.
```

In Java, strings are **not** byte arrays ... more in Lecture 7

Socket Communication 5/5

- Implementation, client side
  - For a web application, suffices to implement the server
  - The web browser plays the role of the client
  - We can also test via simple communication programs, e.g.

    telnet <host> <port>

    Establishes a communication channel to the given machine and port

    You should also try this when you work on ES6, to check if your basic server loop works

Hypertext 1/7

- HTTP = Hypertext Transfer Protocol
  - Used by the browser to communicate with (web) server
  - The typical request looks as follows:

    GET /search.html HTTP/1.1 ...

    /search.html = part of URL after the http://<host>:port
  - The typical results is as follows:

    HTTP/1.1 200 OK
    Content-Length: 137
    Content-Type: text/html

    ... the 137 bytes of the content ...

    Note: HTTP demands that newlines are encoded as **\r\n**

Hypertext 2/7

- HTTP = Hypertext Transfer Protocol
  - There are many more request types ... for example:

    POST (instead of GET)

    For longer requests, that are not sent as part of the URL
  - And many more headers ... for example

    HTTP/1.1 404 Not found

    To indicate that the requested resource does not exist

    HTTP/1.1 403 Forbidden

    To indicate that this resource is a no no for you

    For ES6, you must implement 404 ... 403 is optional

Hypertext 3/7

- Content Types
  - Standard names for the different types of content sent across the internet

    Also called MIME = Multipurpose Internet Mail Extensions
  - Examples

text/plain	plain text
text/html	HTML ... see slides 15 + 16
text/css	CSS ... see slide 17
application/javascript	JavaScript ... see slides 19 – 26
application/json	JSON ... see slide 25
image/png	PNG image

Hypertext 4/7

- Browser Development Console
  - Extremely useful for debugging web applications, or in general to understand better what is going on

    Chrome **F12** / Ctrl+Shift+I

    Firefox **F12** / Ctrl+Shift+I

    Internet Explorer **F12**
  - Important sections for today and ES6:

    Network: requests sent and results received

    Elements: elements of the HTML page ... see next slides

    Console: output from the JavaScript ... see slides 18 – 26

Hypertext 5/7

- HTML = Hypertext Markup Language
  - Language for specifying the content of a web page
  - XML-like language, general structure:

```
<html>
  <head>
    ... meta information + includes ...
  </head>
  <body>
    ... contents of the page ...
  </body>
</html>
```

Hypertext 6/7

- HTML
  - Example tags for the <head>...</head> section:

    <link rel="stylesheet" type="text/css" href="..."/>
    <script src="..."></script>

    Include style information and code ... see coming slides

  - Example tags for the <body>...</body> section

...	Level-1 heading
<p> ... <p>	A paragraph of text
<input> ... </input>	Input field
...	Arbitrary "logical" section

Hypertext 7/7

- CSS = Cascading Style Sheets
  - Specify style information (layout, font, color, etc) independent from the contents of the page
  - Has its own (simple) syntax ... for example, all level-1 headings in blue and boldface

    h1 { color : blue; font-weight: bold }
  - When several rules apply to same element, the "most specific" rule wins

    Hence the "cascading" ... used a lot for larger web sites

    For ES6, make some non-trivial changes to the CSS from the lecture, for a more pleasing appearance

JavaScript 1/9

- Motivation
  - A language that runs as part of a web page

    Can do (almost) arbitrary computation

    Can do (almost) arbitrary communication

    Can dynamically change the contents of the web page in response to user actions

    Nowadays, there is hardly a web page anymore without JavaScript in it

JavaScript 2/9

- Language features
  - An object-oriented script language, with a syntax similar to Java, hence the name

    Speed similar to Python, when interpreted line by line

    Modern browsers perform just-in-time (JIT) compilation, in order to achieve speeds similar to Java

  - Variables are untyped

```
var x = 1;                          // Scalar value.
var s = "doof";                     // String.
var a1 = [1, "doof", bloed"];       // Array (mixed types).
var a2 = { "yes" : 5, "no" : 3 }    // Associative array.
```

JavaScript 3/9

- DOM = Document Object Model
  - Well-defined scheme for how to address elements in a web page, in particular by JavaScript code
  - For example: get the contents of an element with a particular id on the web page

    In the HTML:

    ```
    <div id="result">NO RESULT YET</div>
    ```

    In the JavaScript:

    ```
    document.getElementById("result").innerHTML = "42";
    ```

JavaScript 4/9

- AJAX = Asynchronous JavaScript and XML
  - Old name for communication between JavaScript in browser and some server elsewhere ... typical code:

```
xhr = new XMLHttpRequest();
xhr.onreadystatechange = function() {
   if (xhr.readyState == 4 && xhr.status == 200) {
      response = xhr.responseText;
      ... process the response ... }}
   xhr.open("GET", "<url>", true);
xhr.send();
```

Much simpler with libraries like jQuery ... next slides

JavaScript 5/9

- jQuery
  - jQuery is a JavaScript library with convenient functions for all the common stuff ... include via

    ```
    <script src="http://code.jquery.com/..."></script>
    ```

  - Usage examples

    ```
    $(document).ready(function() { ... })
    ```

    Execute included code when HTML has fully loaded

    ```
    $("#heading").html("Different text")
    ```

    Change contents of element with id "heading"

JavaScript 6/9

- jQuery
  - Offers a much cleaner separation between static elements (HTML) and dynamic code (JavaScript)
  - For example: do something after each keypress

    Raw JavaScript:

    HTML: `<input id="query" onkeypress="myFct()"/>`

    JavaScript: `myFct() { /* ... code here ... */ }`

    With jQuery:

    HTML: `<input id="query">`

    JavaScript: `$("#query").keypress(function() { ... })`

JavaScript 7/9

- jQuery, communication with server
  - For example: launch GET request and do something with the result:

```
url = "http://" + host + ":" + port + "/?q=" + query;
$.get(url, function(result) {
   console.log("Server replied: " + result);
   $("#result").html(result);
})
```

Note: writing to the console is quite useful for debugging

JavaScript 8/9

- JSON = JavaScript Object Notation
  - The result from a computation is often a complex object, e.g. an array or associative array
  - If sent as a mere string, we need code to parse that string on the JavaScript side
  - JSON is content in the form of ready-to-use JavaScript code ... for example:

    ```
    [ "QUERY", "query" ]
    ```

JavaScript 9/9

- jQueryUI
  - Extension of jQuery for more complex UI elements

    ```
    <script src="https://code.jquery.com/ui/..."></script>
    ```

  - For example, autocompletion from fixed set of strings
  - HTML:

    ```
    <input id="query">
    ```

  - JavaScript:

    ```
    $("query").autocomplete({
      source: [ ... array of strings from
                  which to autocomplete ... ]
    });
    ```

References

- Relevant Wikipedia articles (in order of appearance)

  http://en.wikipedia.org/wiki/Network_socket

  http://en.wikipedia.org/wiki/Hypertext_Transfer_Protocol

  http://en.wikipedia.org/wiki/Internet_media_type

  http://en.wikipedia.org/wiki/HTML

  http://en.wikipedia.org/wiki/Cascading_Style_Sheets

  http://www.w3schools.com/js

  http://en.wikipedia.org/wiki/Document_Object_Model

  http://en.wikipedia.org/wiki/Ajax_(programming)

  http://jquery.com/ http://jqueryui.com/