Databases and the Cloud: The Notes

Josh Felmeden

2018
December

Contents

1 The Internet

End systems are connected via the **communication links** that consist of the different types of physical media. Usually, the end systems are not directly attached by a single link, but rather they are attached through a router.

There are two kinds of host: *clients* and *servers.* A program or machine that responds to request and others is called a **server** while a program or machine that sends the requests to the server is called a **client**.

The internet is made possible by the development, testing, and implementation of the *internet standards.* They are developed by the Internet Engineering Task Force (or the IETF). Their documents are known as RFCs (request for comments). There are a number of protocols, such as TCP, IP, HTTP, and SMTP (this one is used for emails). There are more than 2000 RFCs.

1.1 Protocols

A **protocol** is a set of rules that govern the communication to ensure a standard of communication. It also consists of messages sent and actions taken in response to replies or other such events.

A simple protocol could be where one machine sends a message (called a *request*) and another machine replies with a response. This can then be repeated.

1.2 Internet Layers

- HTTP
  - Makes request
  - Reads and handles the response
- TCP
  - Breaks data up into packets
  - Puts the packets back in order and reassembles messages
- IP
  - Attaches to and from addresses to each packet
  - Reads and groups packets based on the address
- Physical internet
  - Send bits to local routers
  - Receives bits and assembles into packets

1.3 HTTP: Hyper text transfer protocol

What's the difference between the web and the internet? Well, the internet is the computer network itself (or the whole infrastructure): while the web (or the world wide web) is an application that runs on that infrastructure.

It's probably the most common application protocol that there is on the web (but there are others like video streaming and FTP and the like). Right now, there's a version 2.0, but we'll be focussing on version 1.1 here.

1.4 Crud

CRUD is an acronym for the basic operations that can be carried out on data.

- Create
  - The create interaction creates a new resource in a server assigned location. The create interaction is performed by a HTTP POST method.
- Read
  - The read interaction accesses the current contents of a resource. The interaction is performed by a HTTP GET method
- Update
  - The update interaction makes a whole new version for an existing resource (or makes a new one if there isn't one)
- Delete
  - The delete interaction deletes an existing resource

1.5 Structure

HTTP is *line-based* and each line ends with a **carriage return line feed** (CR LF). In it, there is a header and a method.

1.6 Status codes

There are some cases where an interaction does not go well. The response from a server can be a number, and the first digit informs you of the nature of the error.

1.7 URLs

The internet needs to have addresses. It needs to know the addresses of both the client and the server. The URL (**uniform resource locator**) tells you where some resource is. A resource is an *address*.

2 Developing web pages

2.1 Markup

Historically, marking up a paper manuscript was done by editors to show authors how to revise their manuscripts. The markup was done in *blue pen* to make it distinguishable from the manuscript text.

In electronic documents, **tags** are used to make the markup distinguishable from the content. A markup language is used to annotate a document.

2.2 HTML

HTML (or hypertext markup language) consists of a fixed set of *tags* that describe how information should be displayed. For example:

```
<p> This is some text </p>
<h1> This is some header </h1>
```

The Browsers do not display the HTML tags, but they use them to render the content of the page.

HTML5 is different from HTML because it's simpler, but also **semantic** (which means that some of the tags describe what the data means as well has how it should be displayed). It also has some more features.

Example HTML5:

```
<!DOCTYPE html>
<html lang="en">
    <head>
        <meta charset="utf-8" />
        <title>My title</title>
    </head>
    <body>
        content
    </body>
</html>
```

Tags have to be nested too.

A block-level element always starts on a new line and takes up the full width available. Conversely, an inline element doesn't start on a new line and it only takes up as much width as is necessary.

The <div> tag is a block level tag that has no specific meaning. This is OK to use for layout purposes, but you should not use it as a replacement for something that should be a semantic tag. Because the semantic tags are mostly a new addition to HTML5, older frameworks used <div> all over the place to structure the pages.

2.2.1 Attributes

In this example:

```
<p id="today">
    28 September
</p>
<p class="info">
    lecture 2
</p>
<p class="info">
    QB 0.18
</p>
```

2.2.2 Links

Almost anything can go inside a <a> tag: text, images, other HTML elements. The href could be a full URL, or it can be relative to the current page.

The main issue with HTML is that you need to structure your web pages really carefully because it's going to be viewed on all kinds of devices and browsers.

Here are some basic rules:

1. Use lower case element names
2. Close all your elements (you don't need to close them in HTML5 but do it anyway)
3. In HTML5, it's optional to close the empty statements, but do it anyway.
4. HTML5 allows the mixing of uppercase and lowercase names, but just use lowercase because it looks nicer and it's easier to write.
5. HTML5 allows attribute values without quotes but again, it's bad because it looks ugly
6. ALWAYS add the alt attribute to images, because if, for some reason, the image can't be displayed, you need some alternate text to display. It's also used for people using screen readers.
7. In HTML5, the html and body tag can be omitted, but, again, it's **bad**. It can crash some XML software.
8. To ensure that everything is interpreted and has correct search engine indexing, the language AND the character encoding should be defined as early as possible.
9. Don't use absolute pixel width measurements

2.3 Forms

The form tag is used for things like buttons, text boxes, etc. It has input types of things like:

- Button
- Month
- Number
- Text
- Password
- Color
- Date
- ...

The **action** attribute defines the action to be performed when the form is submitted. Normally, the data from the form is sent to a web page on the server when the user clicks on the submit button. For example:

```
<form method="post"
action="/action_page.php">
</form>
```

In this example, the data is sent to a page on the server called "/action_page.php". This page contains a script that will handle the form data such as storing it in a database.

There are two (2) methods to send form data, **GET** and **POST**. In HTML5, browser forms support them both. GET places form data in the URL parameters by default (GET/search?query=pancakes), while POST sends the data in the HTTP request body. There are fewer limitations and it's more secure because the data is not visible in the URL.

2.3.1 Validation in forms

If you use type="number", then it won't let you type in letters. It you use required, then the browser won't let you submit if the field is empty.

Place holder is text that can be displayed while the field is empty. It's NOT a label.

2.3.2 Buttons

Buttons can have these types:

- Submit (default)

- Reset: reset all form fields
- Button: do nothing by default (use this if you're using JS).

2.4 CSS

Use HTML for the structure, and then CSS for the styling. This includes layout, appearance, and some behaviours. You can customise ANYTHING. If you want emphasised words to be underlined, then by golly