Hi guys

Download either

Notepad++
Atom
Brackets

Or you can use any code editor you are familiar with.
We start at 7:10

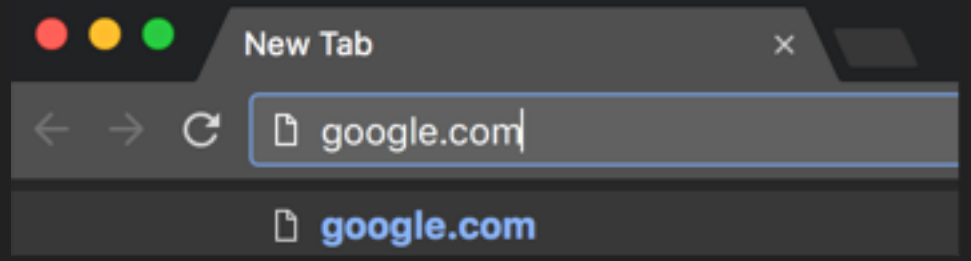
3DC Web Development Series

Part 1 – Introduction to the Web & the Hyper Text Markup Language

Theory of the Internets

Sample ISTD exam question

Describe and explain what happens when you type “google.com” into your web browser and hit [Enter]. (30 marks)



Theory of the Internets

GitHub – “What happens when you type google.com into your browser”

https://github.com/alex/what-happens-when

(the full answer is so long I can’t even save a full page screenshot of it)

This is a collaborative process, so dig in and try to help out! There are tons of details missing, just waiting for you to add them! So send us a pull request, please!

This is all licensed under the terms of the Creative Commons Zero license.

Read this in 简体中文 (simplified Chinese) and 한국어 (Korean). NOTE: these have not been reviewed by the alex/what-happens-when maintainers.

Table of Contents

The "g" key is pressed

The following sections explain the physical keyboard actions and the OS interrupts. When you press the key "g" the browser receives the event and the auto-complete functions kick in. Depending on your browser's algorithm and if you are in private/incognito mode or not various suggestions will be presented to you in the dropbox below the URL bar. Most of these algorithms sort and prioritize results based on search history, bookmarks, cookies, and popular searches from the internet as a whole. As you are typing "google.com" many blocks of code run and the suggestions will be refined with each key press. It may even suggest "google.com" before you finish typing it.

The "enter" key bottoms out

To pick a zero point, let's choose the Enter key on the keyboard hitting the bottom of its range. At this point, an electrical circuit specific to the enter key is closed (either directly or capacitively). This allows a small amount of current to flow into the logic circuitry of the keyboard, which scans the state of each key switch, debounces the electrical noise of the rapid intermittent closure of the switch, and converts it to a keycode integer, in this case 13. The keyboard controller then encodes the keycode for transport to the computer. This is now almost universally over a Universal Serial Bus (USB) or Bluetooth connection, but historically has been over PS/2 or ADB connections.

In the case of the USB keyboard:

- The USB circuitry of the keyboard is powered by the 5V supply provided over pin 1 from the computer's USB host controller.
- The keycode generated is stored by internal keyboard circuitry memory in a register called "endpoint".
- The host USB controller polls that "endpoint" every ~10ms (minimum value declared by the keyboard), so it gets the keycode value stored on it.
- This value goes to the USB SIE (Serial Interface Engine) to be converted in one or more USB packets that follow the low level USB protocol.
- Those packets are sent by a differential electrical signal over D+ and D- pins (the middle 2) at a maximum speed of 1.5 Mb/s, as an HID (Human Interface Device) device is always declared to be a "low speed device" (USB 2.0 compliance).
- This serial signal is then decoded at the computer's host USB controller, and interpreted by the computer's Human Interface Device (HID) universal keyboard device driver. The value of the key is then passed into the operating system's hardware abstraction layer.

In the case of Virtual Keyboard (as in touch screen devices):

Theory of the Internets (simplified)

databases

Search engines

DNS Server

Hey 10.11.87.1, do you serve a website named "google.com?"

10.11.87.1
Google's public web server

Where is google?

Google is at 10.11.87.1

HTTP Request

HTTP Response

Yes, I do, and it looks like this:

Web browser

Parsing & Rendering

You enter "google.com"

Theory of the Internets (simplified)

Backend Development

databases

Search engines

10.11.87.1
Google's public web server

DNS Server

Frontend Development

HTTP Request

HTTP Response

You enter "google.com"

Web browser

Parsing & Rendering

Web Dev is Huge

Frontend Development

- Concerned with the **user-facing** parts of the web application
- "Application" *is* the HTML document
- Look and feel of the webpage
- Technologies used: HTML, CSS, JS

Backend Development

- Application receives and processes the HTTP request, and crafts the HTTP response (HTML document) to return to the client
- Talks to databases and other services
- Technologies used: Python, NodeJS, PHP, Ruby, SQL…. (anything that runs on a server)

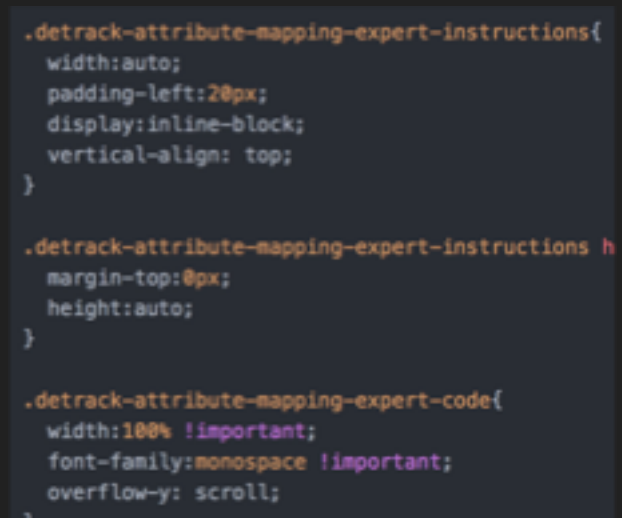
The Holy Trinity of Frontend Development

HTML

- The underlying structure of a webpage. Tells your browser what elements and contents there are on the webpage.

CSS

- Formatting instructions. Tells your browser how to position, colour and resize your elements on the webpage.



JS

- The language of the web that breathes life into your webpages. Tells the browser to execute code during certain events (mouseclick etc)

HTML Only

Just text and some images.

Ugly AF (like me and my slides)

SUTD

- About Us
  - Vision and Mission
  - Next Phase of Growth
  - Accreditation
  - Collaborations
  - People
  - Achievements
  - News and Events
  - Careers with SUTD
  - Procurement
  - Giving
  - Building Gender Diversity
  - Contact Us
- Admissions
  - Undergraduate
  - Graduate
- Education
- Research
- Student Development
- Campus Life
- Global

- Prospective Student
- Student
- Faculty/Staff
- Partner
- Enter keyword... Search

Menu Search

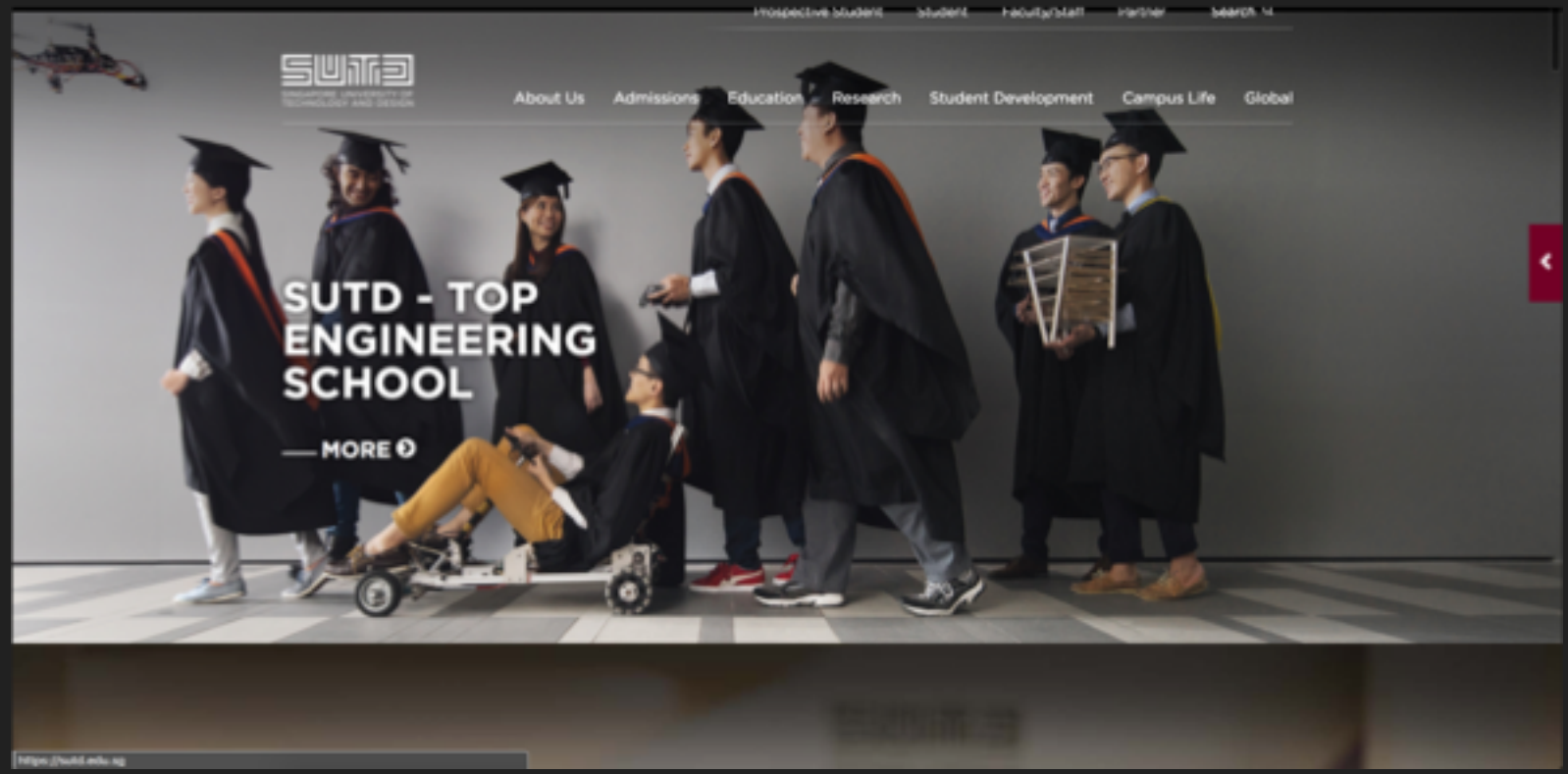
Search by keyword...

- About Us
- Admissions
  - Undergraduate
  - Graduate

HTML + CSS

Nicely formatted text

Looks much better, but the webpage is completely static.



HTML + CSS + JS

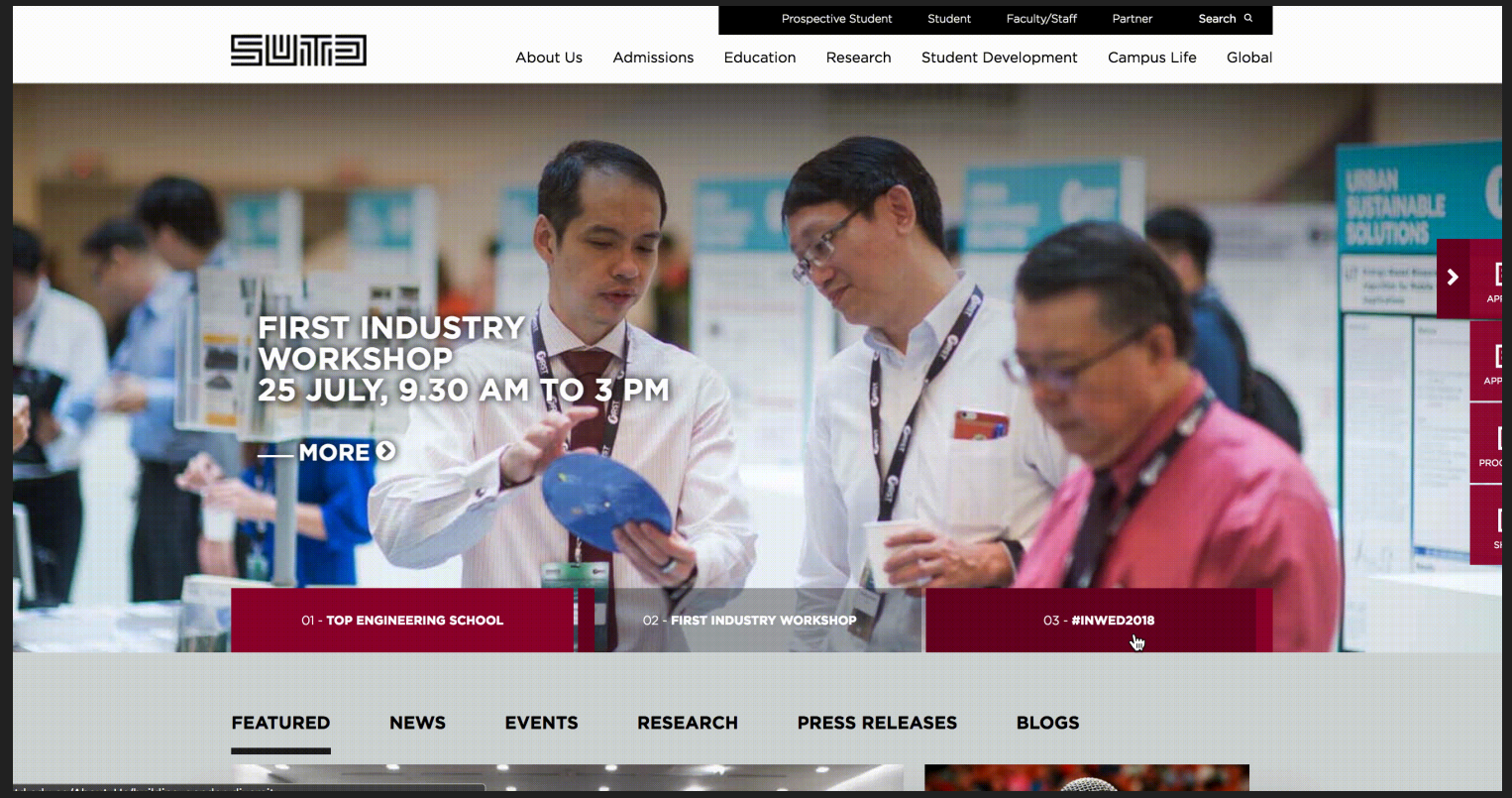
Interactive elements!

Dropdown menus!

Moving Image Galleries!

Everything is super flashy!

(ok not SUTD but ya)



Let’s get started!

HTML Syntax

- HTML is written in *tags*, similar to XML
- Tags can contain content, or contain other tags.
- Open a tag like this:
  - `<tagname> Content here </tagname>`
- Once opened, tags must be properly closed. (some exceptions apply)
- HTML provides a large number of predefined tags to create your webpages. Each tag creates an *element* (also called a *node*).
  - `<p> This creates a paragraph. </p>`
- When tags go inside another tag, the inside tags are called *children*, the outside tags are called *parents*.
  - `<p> I can put another <span> element </span> inside my paragraph tag!</p>`

HTML Structure

- Proper HTML documents must, first, be wrapped in a `<html>` tag preceded by a `<!DOCTYPE html>` The first child in that tag should be the `<head>`, then the `<body>`.
- The header is where you put your CSS and JS and some other preliminary stuff.
- The body is where your actual webpage content goes.

Open your code editor, type the following, save as a .html file and open it in your web browser.

```
<!DOCTYPE html>
<html>
  <head>
    <title>This is a title</title>
  </head>
  <body>
    <p>Hello world!</p>
  </body>
</html>
```

Protip: whenever you put a tag inside another tag on a newline, you should indent the following line with a TAB space to keep things readable.

Side note: the DOCTYPE tag is special and doesn't need to be closed.

HTML Elements - basic

- `<p>` Paragraph tag. You can insert text in this element and it will appear as a paragraph. It will **occupy the full width** of the parent and **clear some space** above and below itself.
- `<h1>` Header tags for titles and subtitles. Ranges from `<h1>` to `<h6>`, `<h1>` being the largest and `<h6>` being the smallest. You can use these tags to specify sections in your text.
- `<br>` To insert a blank line if you want to create separate stuff, but using CSS is preferred for spacing (next week). **`<br>` is a self closing tag, you do not need to close it.**
- `<span>` Generic inline container for holding text. Unlike `<p>`, it only takes up the **width of its content**, and **will not clear space** around itself unless you use CSS to do so. Any text left outside of any tags will be rendered like a span.

Exercise: Using these four tags, now create a simple page with headers (titles and subtitles) describing yourself or your school or your favourite thing. (10mins)

HTML Elements – text formatting

- `<em>` Emphasis tag. Renders text inside the tag as *italics*.
- `<b>` or `<strong>` Bold tags. Renders text inside the tag as **bold**.
- `<pre>` Preformatted text with fixed width `will appear like this`.
- `<sup>` Superscript$^{\text{like this}}$, `<sub>` subscript$_{\text{like this}}$.

Exercise: Using these tags, add some flavour to the webpage you just made. Try combining bold and italics to achieve ***something like this.*** (10min)

A Note on Nesting Elements

- Tags **must** be closed in the same order that they were opened, or else it is not valid and your webpage will break.
- Check if you accidentally closed a tag before it was even opened.
- Most tags can be nested recursively (such as `<span>` in a `<span>`), and you must take care to not prematurely close the earlier tag by accident.

Correct:

```
<p>This is a <strong>bold</strong> statement</p>

<p>This is a <strong>bold</strong> and <em>stylish</em> statement </p>
```

Wrong:

```
<p>This is a <strong>bold</p> statement</strong>

<p>This is a <strong>bold</em><strong> and <em>stylish statement</p>
```

Another Note on Nesting Elements

- Tags like and will format the text all the way until it is closed, so pay attention to the order of the tags.

```
<p>I am <strong><em>bold and
stylish</em></strong></p>
```

is the same as

```
<p>I am <em><strong>bold and
stylish</strong></em></p>
```

But

```
<p>I am <strong>bold and
<em>stylish</em></strong></p>
```

is not the same as

```
<p>I am <em><strong>bold</strong>
and stylish</em></p>
```

HTML Tag Attributes

- HTML allows you to add attributes (properties) to your elements (tags), in the following format:

```
<tagName attr1="value1" attr2="value2">Content</tagName>
```

- Tag attributes must be declared in the **opening tag only.** Attribute values **must be enclosed in quotes** (either single or double).
- You can give names and ids to your elements (which will be used by CSS and Javascript to find your elements), or add CSS directly to your elements to change properties such as font, colour and background.
- There are universal HTML attributes such as `class` and `id` that will work on any tag, while there are some that can only be used on specific tags.
- Some HTML tags require attributes in order to function.

The <a> tag

- An example is the anchor tag, `<a>`.
- This turns the inside text into a hyperlink or as an anchor depending on how you use it.
- To tell the browser where to bring the user after clicking on the link, you need to use the `href` attribute.

as an anchor

```
<h2><a href="#about-me">About me</a></h2>
```

as a hyperlink

```
<p> I am an undergraduate student at the <a href="https://sutd.edu.sg">Singapore University of Technology and Design</a>, studying World Texts and Interpretations.</p>
```

The <img> tag

- Another example is the <img> tag, which lets you embed images.
- Use the src attribute to specify the image source (either an Internet URL or a local file), then the height and width attributes to adjust the size.

```
<img src="https://sutd.edu.sg/assets/sutd/img/logo-black.png" />
<img src="cat.jpg" width="200" height="200" />
```

- The <img> tag is a **self-closing tag**. You **cannot** nest other tags inside an <img> tag.
- If you do not specify the width and height, it will use the full width and height of the original image.
- If you do not put a URL (http or https), it will look for a local file in the same folder as your HTML file.

Exercise: Add some images and links to your webpage. (5mins)

HTML Lists

- Bullet lists like the ones I have been abusing in these Powerpoint slides can be made with the <ul>, and <li> tags.
  1. creates an "unordered list" like the outer bullet point here, while <ol> creates an "ordered list" like the inner bullet number point here.
- After starting the list with <ul> or <ol>, you then add <li> ("list item") tags ***inside*** the <ol> tag to add items to your list.

```
<p> Things to buy </p>
<ul>
    <li> Eggs </li>
    <li> Shin Ramyun </li>
    <li> Lemon Tea </li>
    <li> My hopes and dreams </li>
</ul>
```

```
<p> The Five Precepts </p>
<ol>
    <li> Do not Kill </li>
    <li> Do not Steal </li>
    <li> Do not Lie </li>
    <li> Oops I ran out of space</li>
</ol>
```

Basic Styling with CSS

- You can insert CSS styles directly into your HTML elements by using the `style` attribute.
- For today, let's just change the colour and the font size of text elements.

- `<p style="color:red"> Everything in this paragraph tag will be red! </p>`
- `<p style="font-size:20px"> Everything in this paragraph is now bigger! </p>`
- `<p style="color:red;font-size:20px"> I can combine styles using a semicolon! </p>`

Hint: To change only some parts of a paragraph text, create a `<span>` element inside your paragraph and add the style attribute to that `<span>` instead of the entire `<p>`.

Final Exercise

- Add a list and try styling your texts.
- And that's it for today!
- Next week we'll cover abit more HTML and then dive straight into CSS to make your webpages pretty.
- Good luck for chem/bio!