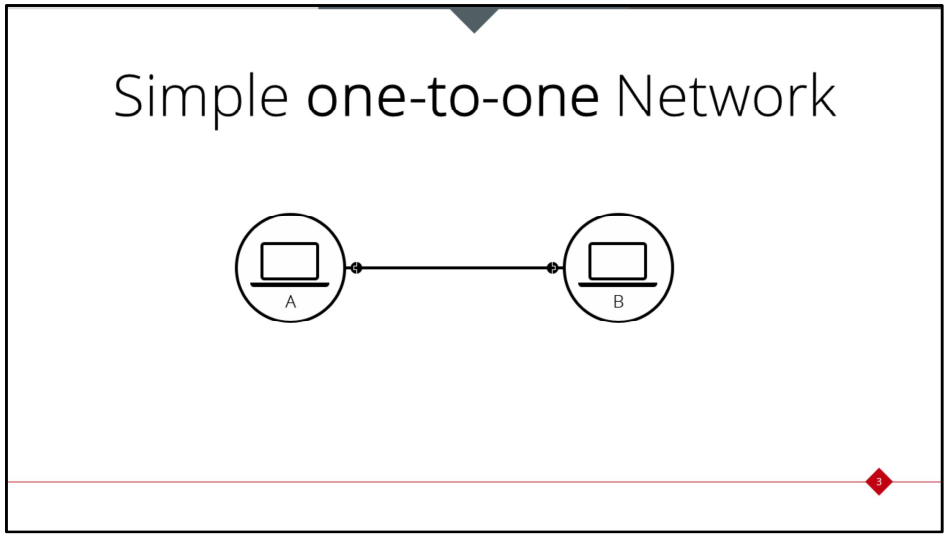
Let's Make Some Website!!!

You will learn about basics of how internet works and build your own website!
It will be a fascinating journey, so be focused and stay on the track, yay!

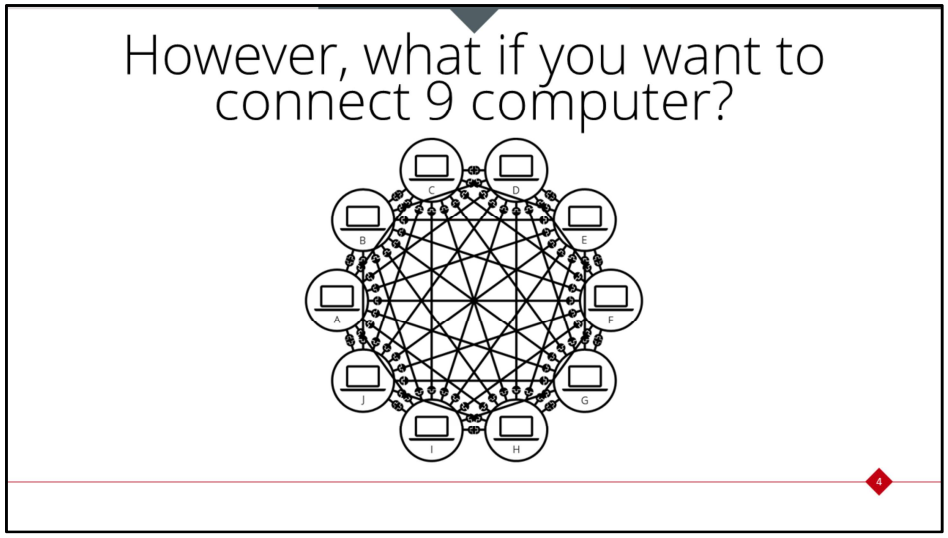
by Divisi Komputer
Himpunan Mahasiswa Elektroteknik
Institut Teknologi Bandung

The Story of Internet

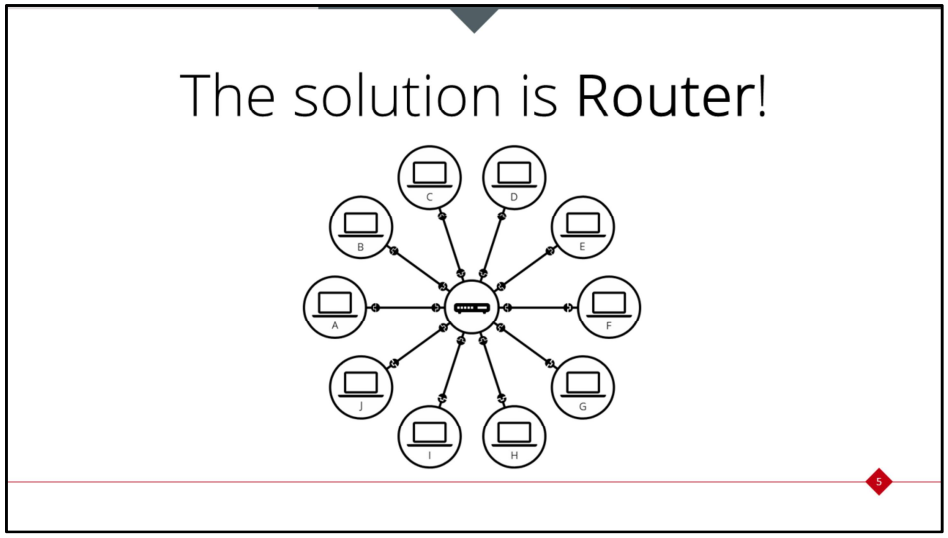
How does the Internet work?



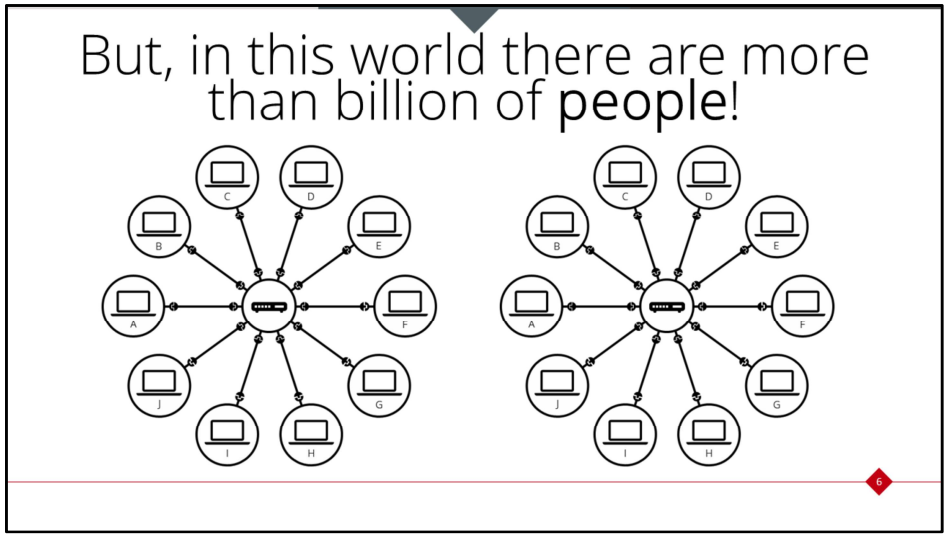
Komunikasi paling sederhana antar dua computer adalah One-to-One Network. Konektivitas antara keduanya bisa aja pake kabel, wireless, Bluetooth, infrared, suara, cahaya, atau apapun itu selama komunikasi diantara kedua computer bisa terjadi.



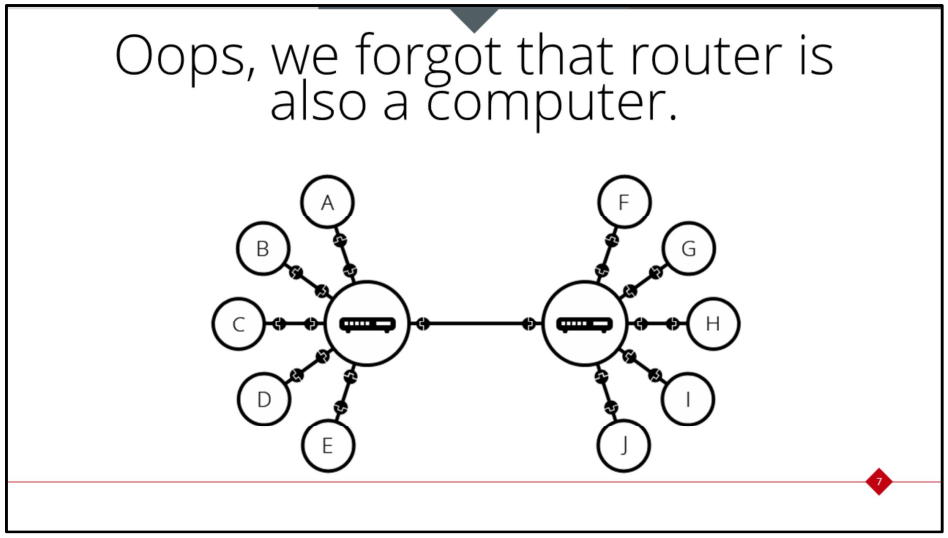
Masalah mulai muncul ketika 9 computer ingin saling terhubung. Wow, kita butuh 45 kabel dengan 9 port untuk setiap computer.



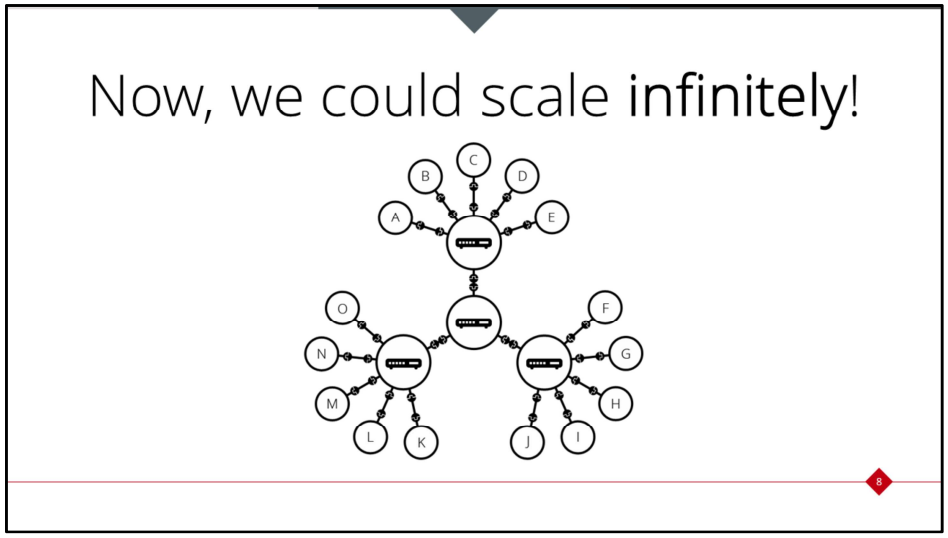
Yay, router muncul sebagai penyelamat. Router adalah computer kecil yang berfungsi sebagai pengatur arah sinyal. Pekerjaan utamanya adalah memastikan pesan yang dikirim akan sampai ke tujuan yang tepat dan tidak salah kirim. Sekarang, kita hanya butuh 10 kabel deh, 1 port untuk masing-masing computer dan 10 port untuk router kita. Yay, dunia menjadi tempat yang lebih indah!



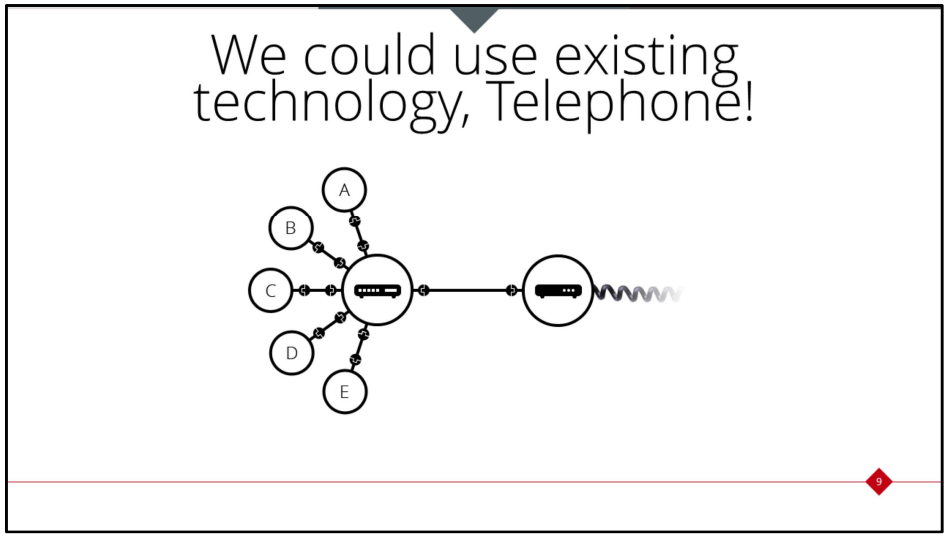
Tapi, problemnya di dunia ini ada lebih dari milyaran orang. Router juga punya batas kesabaran, eh batas port. Kasian kan kalo kebanyakan computer konek ke dia, seberapa pusing apa coba ☹. Bisa aja sih setiap kelompok punya jaringannya masing-masing. Tapi, Kalo orang yang kamu suka ada dijaringan sebelah, gimana cara ngedeketinnya dong? ☹



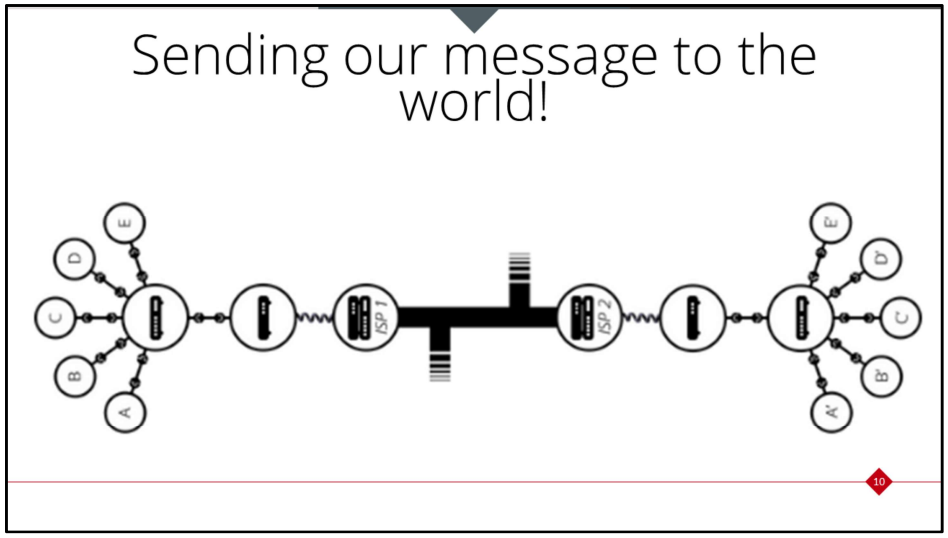
Oia juga, router kan computer, yauda suruh temenan (konek) aja. Sekarang kamu bias tuh deketin doi via dua router. Cieee.



Sekarang, kita bisa scaling hubungan kita ke orang banyak deh! Nah struktur gini tuh uda deket banget sama definisi Internet. Tapi, kita lupa satu hal. Kita bikin jaringan kita untuk tujuan kita sendiri, orang lain juga bikin jaringan untuk tujuan mereka. Tapi, ga mungkin juga kita konekin jaringan kita ke seluruh dunia kan? Butuh kabel berapa panjang coba?

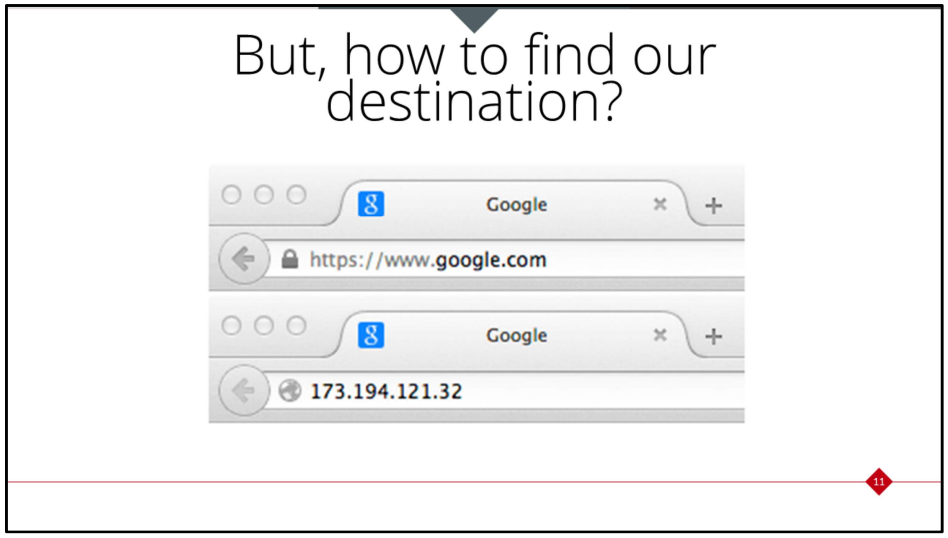


Nah, dengan memanfaatkan jaringan yang sudah ada sebelumnya. Kita bisa membangun jaringan yang lebih keren lagi. Disini, muncul pahlawan baru bernama **Modem.** Pahlawan kita ini mengonversi informasi dari jaringan kita menjadi informasi yang dapat dibawa melalui infrastruktur telepon dan sebaliknya.



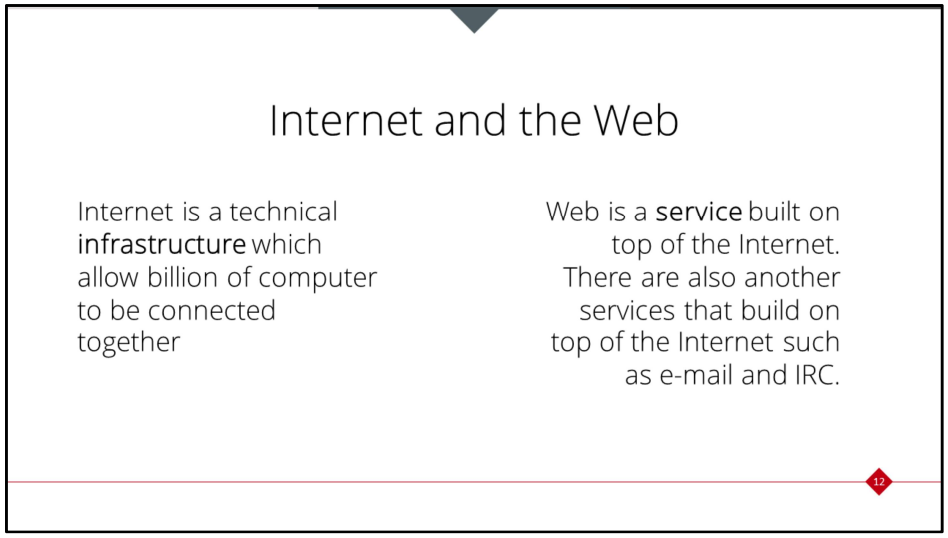
Nah, jaringan telepon kita terkoneksi dengan jaringan luar dengan melalui *Internet Service Provider* (ISP). ISP adalah perusaan yang mengatur router special yang menghubungkan seluruh dunia dan dapat mengakses router special milik ISP lainnya. Pesan kita dapat melewati jalur ini dan sampai ke tujuan deh!

Sederhananya, Internet teh cuma jaringan begini tapi banyak aja.



Pesan kita bisa sampai ke computer tujuan lewat sesuatu bernama IP (Internet protocol) address. Biasanya suka ada tuh di pc kalian kaya 192.168.2.10
Misalnya.

Tapi, karena kita manusia. Kita ga suka inget angka, kecuali tanggal ulang tahun doi sama sisa saldo di bank? (eh). Jadi kita bikin alias dari IP address itu namanya *Domain Name*. Nah, contohnya google.com adalah domain untuk IP address 173.194.121.32 (Contoh ya). Yay kita bias kemana-mana pake nama aja sekarang!

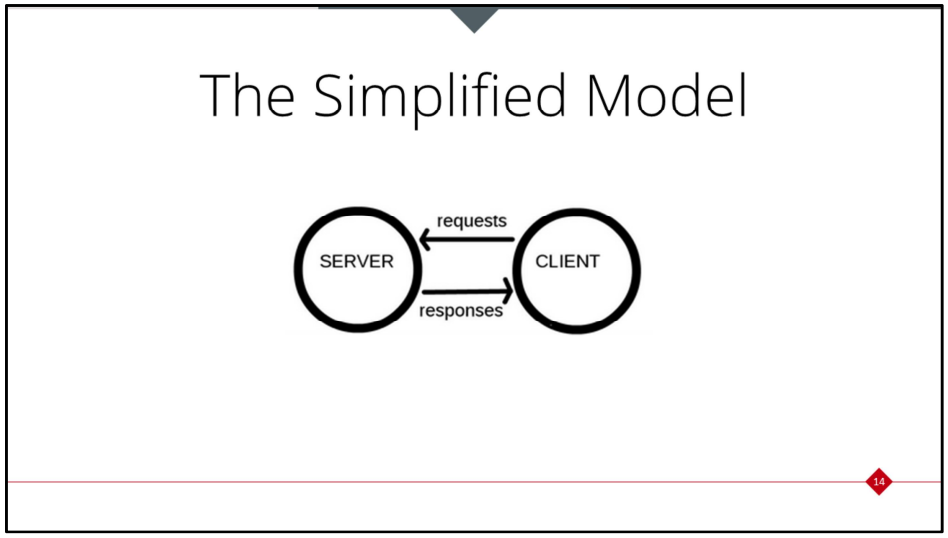


Jadi, dua hal tersebut beda ya gengs!

The Story of Web

How does the Web work?

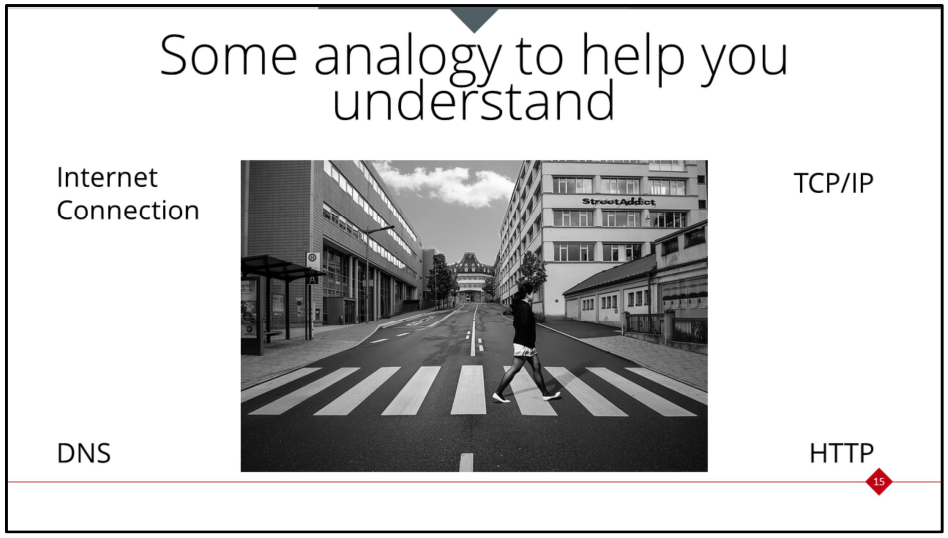
13



Komputer yang terhubung ke web, umumnya dimodelkan sebagai klien dan server.

Klien adalah pengguna web umum yang mengakses web menggunakan gawai yang terhubung dengan internet via web browser. Server adalah computer yang menyimpan halaman web, situs, atau aplikasi. Ketika klien meminta akses ke halaman web, situs, atau aplikasi, server mengirimkan salinannya ke klien agar dapat ditampilkan pada browser klien

Umumnya, model *request* and *response* sering digunakan dalam model web.

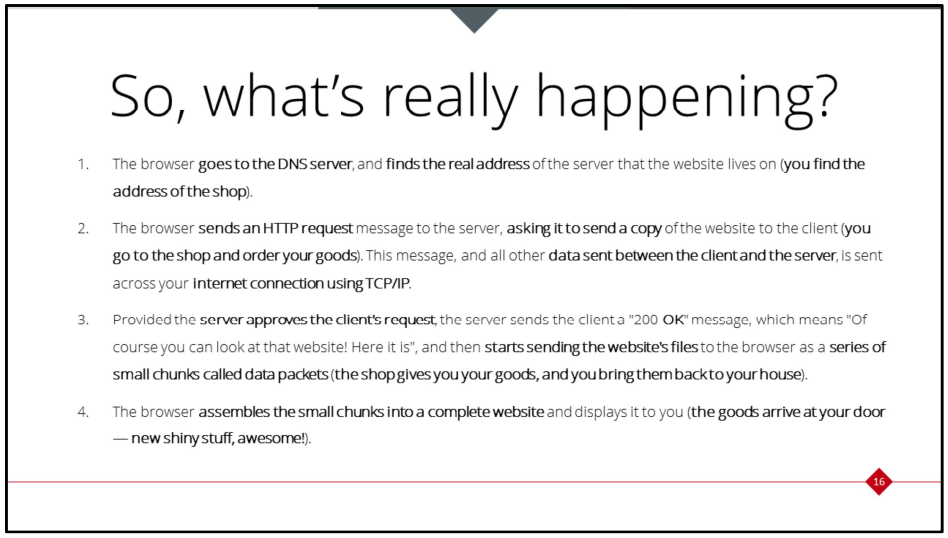


Bayangkan aja kalo Web adalah sebuah jalan. Kalian adalah klien yang datang dari rumah kalian di sebelah kiri jalan dan toko yang kalian tuju adalah server yang ada di kanan jalan.

Selain kedua hal tersebut, kalian butuh:

1. Koneksi internet, yang ngasih kalian akses untuk mengirim dan menerima data dari web. Sederhananya semacam zebra cross atau jalan.
2. TCP/IP, adalah sebuah protocol yang ngasih tau gimana sih cara melakukan sesuatu di web. Contohnya via Mobil, sepeda, atau jalan kaki pasti punya jalur yang beda dan prosedur belanja yang beda (Inget aja McD ada *drive thru* ada yang ngantri konvensional)
3. DNS, sederhananya kaya phone book. Mau tau nih McD ada dimana. Nah DNS ngasih alamat jelasnya (IP Address kalo disini)
4. HTTP, Protokol aplikasi, analoginya Bahasa yang kamu pakai buat mesen di toko tersebut.
5. Selain itu, ada juga file tambahan kaya kode html css sama js dan asset kaya video, gambar, music.

Selain klien dan server. Kalian harus



Satu-satu jelasinnya wkwk

More about DNS and Packets

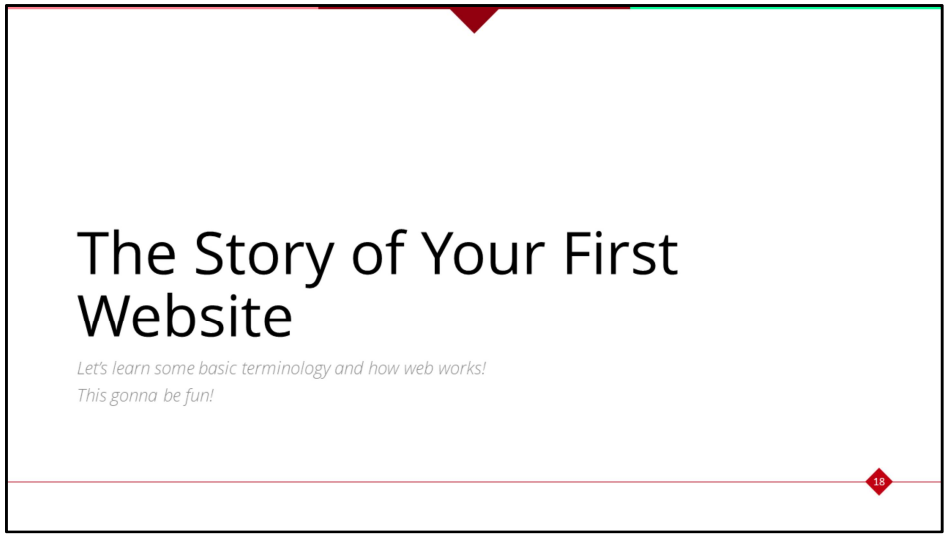
DNS or Domain Name Server is invented because **we don't want to remember all those number** of every website.

DNS would **provide us the IP address** of our favourite website. Cool Right?

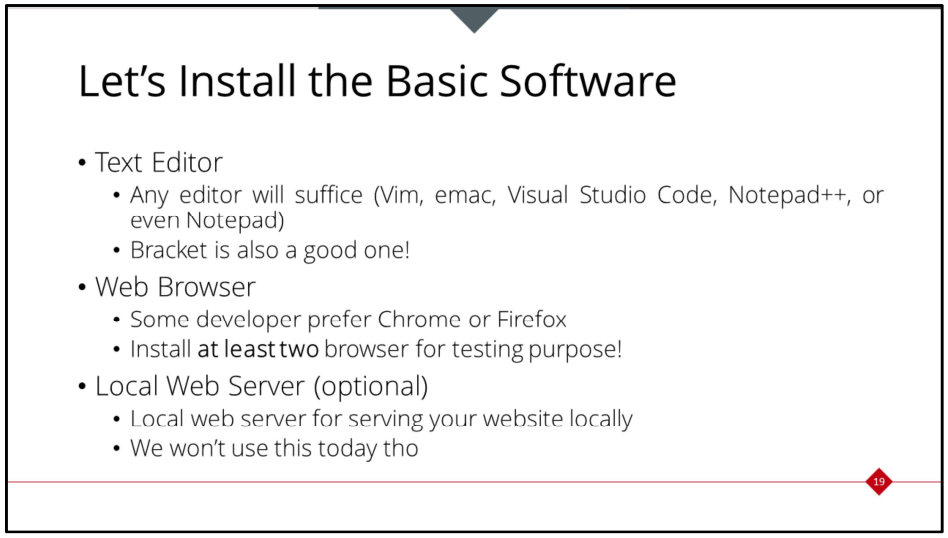
Packets is a **format of sent data** from server to client.

When data is sent across the Web, **data is sent as small chunk**. This will **allow many user** to use the Web at the same time.

If the data is sent as big chunk, only one user can access the web ☹



Sesi Workshop

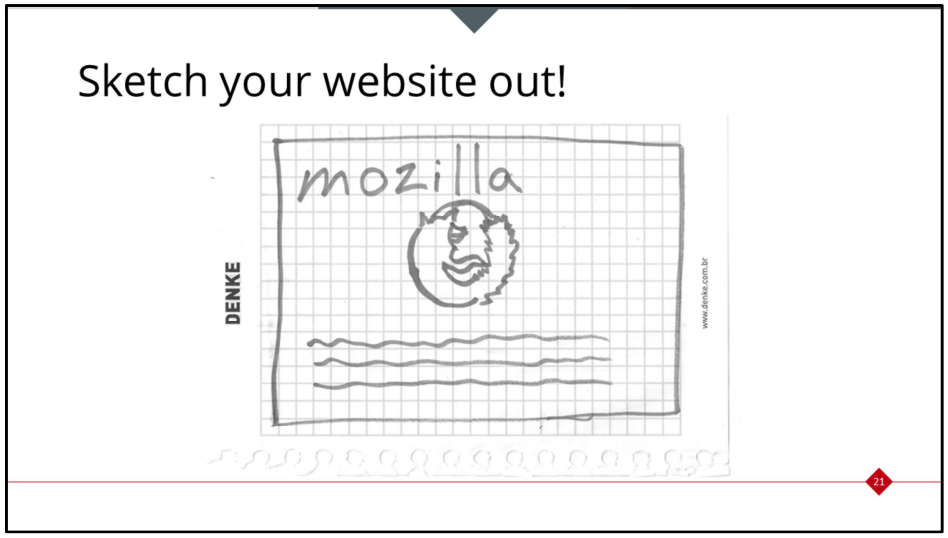


Cukup deskriptif lah ya

Next, plan what will your website look like?

Answer these Question:

1. What is your website about?
2. What information are you presenting on the Subject?
3. What does your website look like?



Pada proyek besar, umumnya desain sketsa kasar seperti inilah yang kemudian didigitalisasi dan dibuat mockupnya. Tim desain proyek mencakup tim UI dan UX.

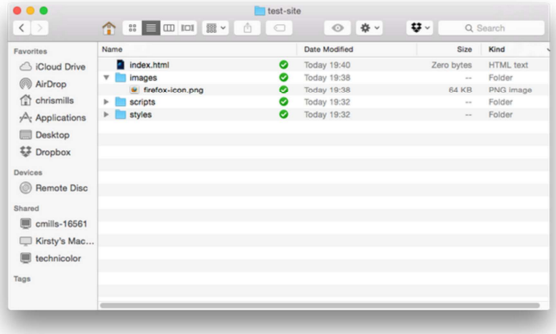
Choosing your Assets!

- Text
  - The paragraph and sentences should be as professional as possible.
  - Make the information as brief as possible, don't confuse reader!
- Theme color
  - Choosing color is one of web designer job, sometimes it will be hard but the other time it will be easy.
- Images
  - You could search for stock images online. But, becareful with copyright!
- Font
  - The font also decide how your website aestethic and readability, so choose carefully!

Organize the file structure!

- Project Folder
  - **Choose a folder** to put your projects such as '**web-project**'
  - Inside the folder, create another folder for your first website '**test-site**'
- Casing and Spacing
  - **Use lower case and hypen (-)** for the file name because usually the web server is *case sensitive*. This will reduce the hassle of remembering which letter is upper case (MyFileName.html).
- Structure
  - **index.html:** Usually this will contain your homepage content
  - **images** folder: This will contain all your images for the website
  - **styles** folder: This will contain CSS code used to style your content
  - **script** folder: This will contain JS code to add some interactive function

The structure should look like this



HTML
The Skeleton of Our Website

Finally, no more story please, let's get into the real thing!

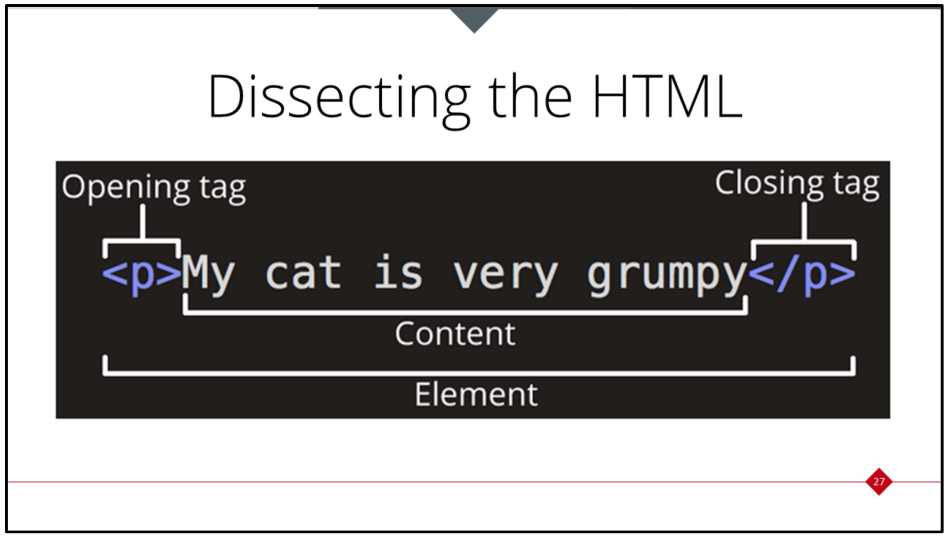
25

What is HTML

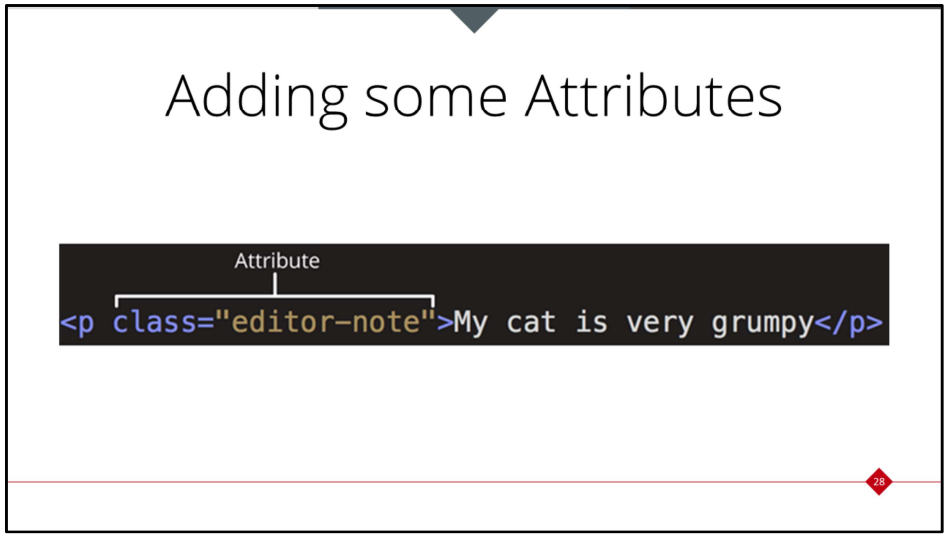
- HTML is **not** a programming language!
- HTML consist of several element that structures your webpage.
- For, example we want to display this as one stand line

  My cat is very grumpy
- In HTML, we could specify the sentence as paragraph by enclosing it in paragraph tags

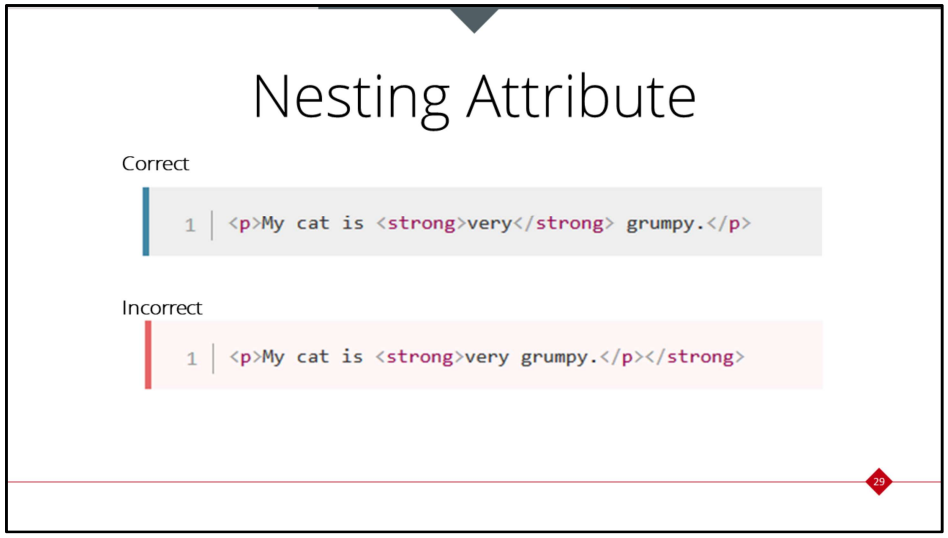
   My cat is very grumpy



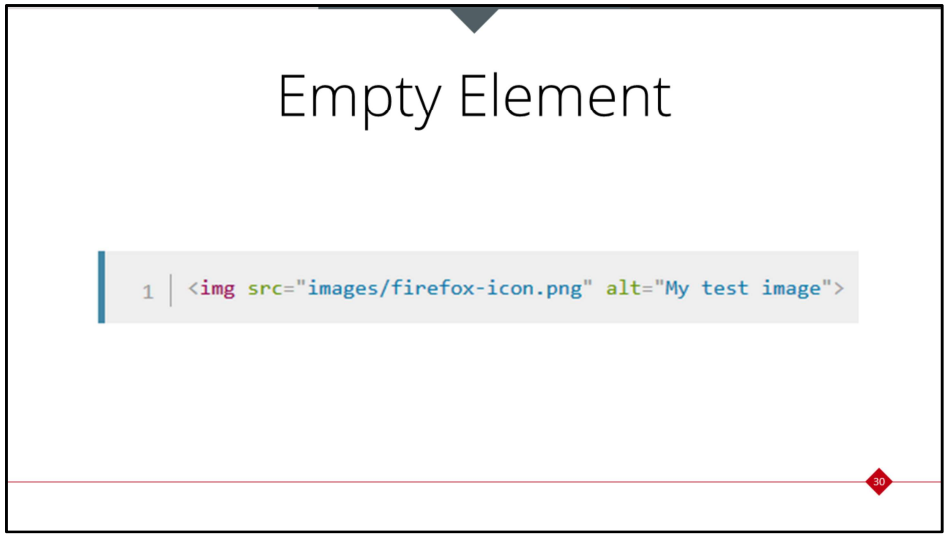
The opening tag: berisi nama elemen, menandakan kapan elemen dimulai (dalam kasus ini paragraph dimulai)
The closing tag: sama dengan opening tag, tapi ditambah *forward slash*, menandakan kapan elemen berakhir. Biasanya pemula suka lupa kasih closing tag, hasilnya ya jadi lucu.
The content: This is the content of the element, which in this case is just text.
The element: The opening tag, the closing tag, and the content together comprise the element.



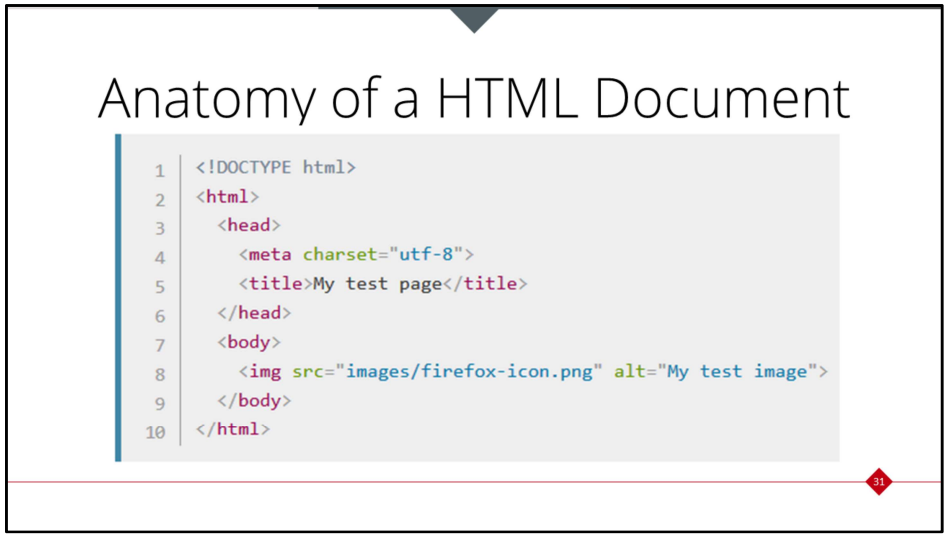
Attribute digunakan untuk menyimpan informasi tambahan tentang elemen yang tidak ingin kita tampilkan pada webpage. Dalam kasus ini, class adalah nama *attribute*, dan editor-note adalah value dari attribute. Attribute dapat digunakan sebagai identifier untuk menargetkan elemen dalam styling nanti



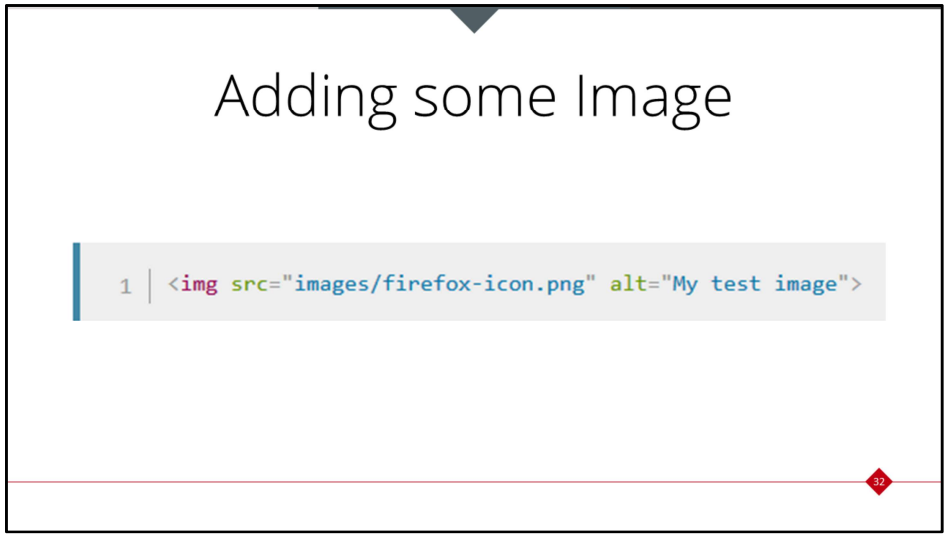
Jangan lupa tagnya harus ngurut nutupnya



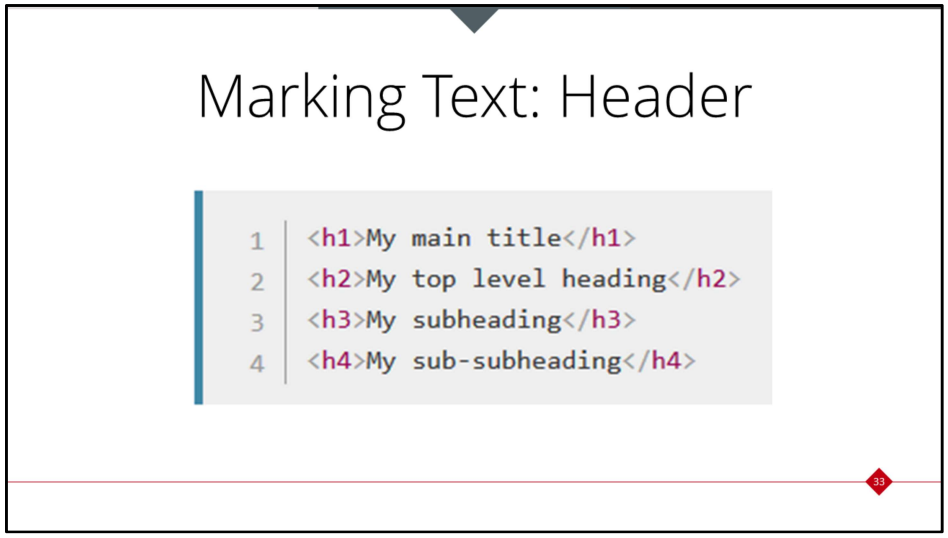
Nah, ada juga yang ga punya content, namanya empty element. Biasanya kalo nampilin media kaya image nih. Penjelasan image ada nanti.



- <!DOCTYPE html> — the doctype: Tambahin aja, konvensi lama di HTML. Ada sejarahnya tapi ga penting-penting amat.
- <html></html> -- Biasanya disebut root element, seluruh sintaks html kita harus ada di dalam tag ini
- <head></head> -- Container dari seluruh element html yang bukan bagian dari konten yang ingin disajikan pada web. Contohnya metadata kaya page description, tag, script, dll
- <meta charset="utf-8"> -- Elemen ini ngatur supaya dokumennya pakai konvensi UTF-8. Salah satu konvensi juga. Soalnya kebanyakan karakter sudah include pakai ini.
- <title></title> -- Elemen ini ngatur judul halaman
- <body></body> -- Elemen ini isinya konten yang pengen disajikan, contohnya tulisan, image, video, dll



Tag ini nampilin image dari src
Pathnya relative terhadap posisi file html ini.
Alt (alternative) adalah tag yang bakal keliatan kalo image dihover atau dibacain buat orang visual impairment screen reader.
Alt text juga bakal muncul kalo imagenya gagal ke load. Jadi, tulis alt text yang cocok yaaa!

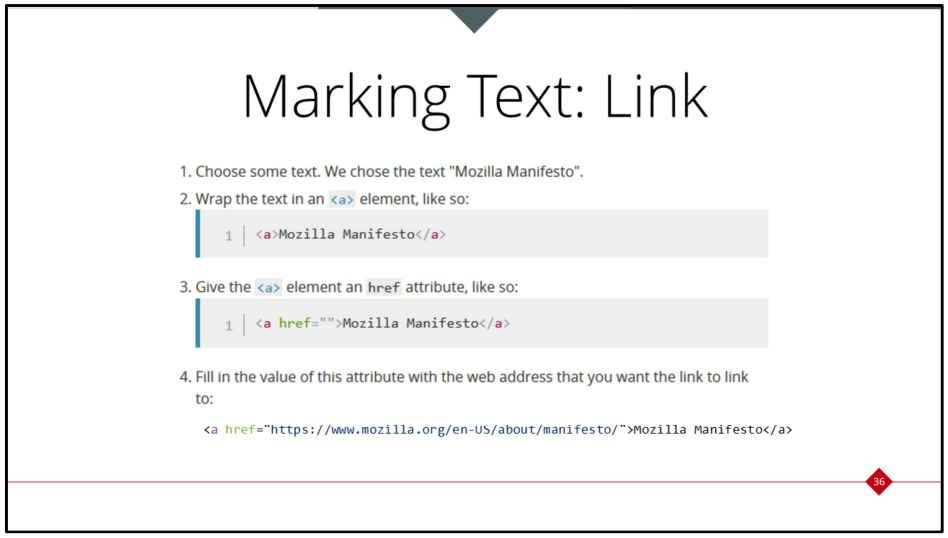


Maksimal header 6.

Marking Text: Paragraph

```
1 | <p>This is a single paragraph</p>
```

Ada dua macam list ordered <ol> sama unordered <ul>.

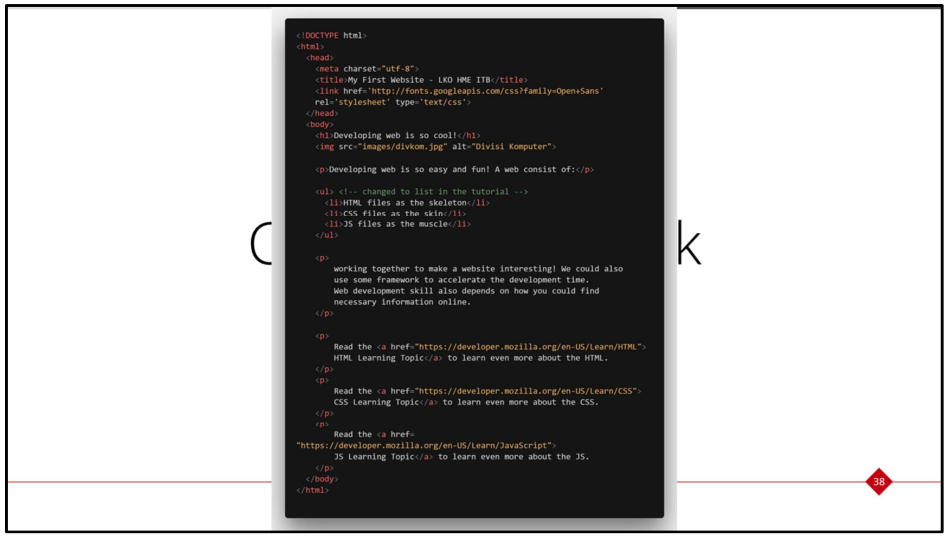


href = remember that it stands for ***h****ypertext* ***ref****erence*.

You might get unexpected results if you omit the https:// or http:// part, called the *protocol*, at the beginning of the web address. After making a link, click it to make sure it is sending you where you wanted it to

href = remember that it stands for ***h**ypertext **ref**erence*.

You might get unexpected results if you omit the https:// or http:// part, called the *protocol*, at the beginning of the web address.
After making a link, click it to make sure it is sending you where you wanted it to



href = remember that it stands for ***h**ypertext **ref**erence*.

You might get unexpected results if you omit the https:// or http:// part, called the *protocol*, at the beginning of the web address.
After making a link, click it to make sure it is sending you where you wanted it to

CSS
The Skin of Our Website

Let's style our Website!

39

What is CSS

- CSS is also **not** a programming language!
- CSS consist of several element that style your website.
- For, example we want to display this paragraph as red colored text

 My cat is very grumpy

- In CSS, we could specify the paragraph by selecting it and apply some style.

```
p {
  color: red;
}
```

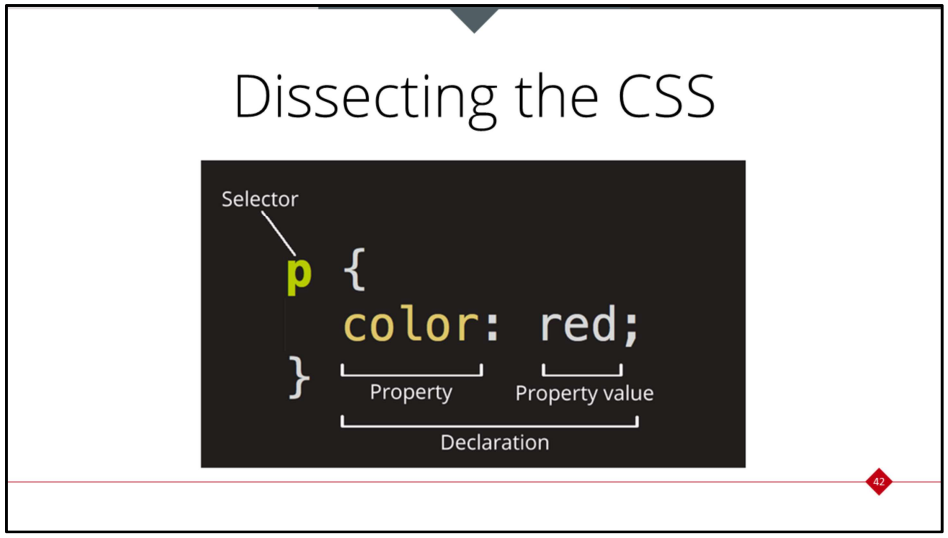
40

What is CSS

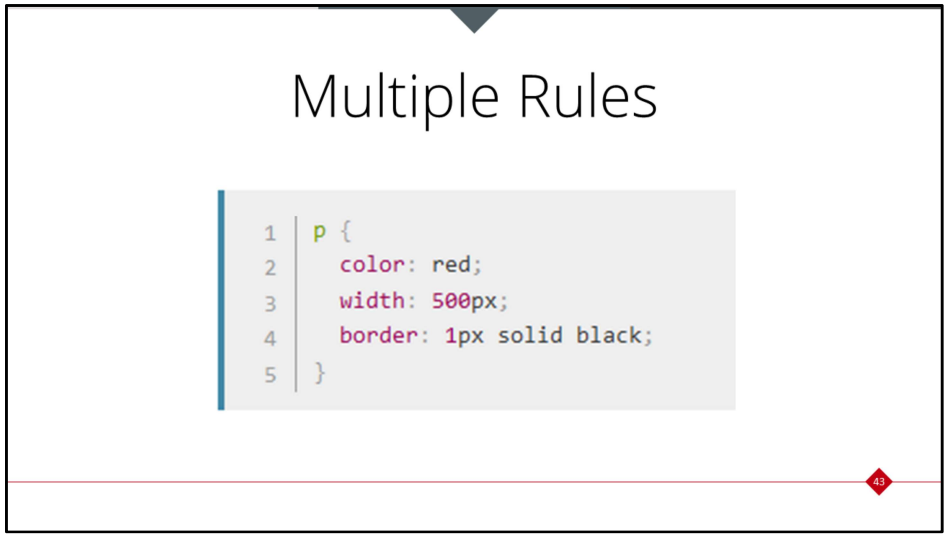
- After that, save the code into `styles/style.css`
- Open your `index.html` and add this line to the <head></head> tag

```
1 <link href="styles/style.css" rel="stylesheet" type="text/css">
```

- Save `index.html` and refresh!



Selector: The HTML element name at the start of the rule set. It selects the element(s) to be styled (in this case, p elements). To style a different element, just change the selector. **Declaration:** A single rule like color: red; specifying which of the element's **properties** you want to style.
Properties: Ways in which you can style a given HTML element. (In this case, color is a property of the <p> elements.) In CSS, you choose which properties you want to affect in your rule.
Property value: To the right of the property after the colon, we have the **property value**, which chooses one out of many possible appearances for a given property (there are many color values besides red).



Also some basic rule:

- Each rule set (apart from the selector) must be wrapped in curly braces ({}).
- Within each declaration, you must use a colon (:) to separate the property from its values.
- Within each rule set, you must use a semicolon (;) to separate each declaration from the next one.

Selecting Multiple Element

```
1  p, li, h1 {
2    color: red;
3  }
```

More Selector

Selector name	What does it select	Example
Element selector (sometimes called a tag or type selector)	All HTML element(s) of the specified type.	`p` Selects `<p>`
ID selector	The element on the page with the specified ID. On a given HTML page, you're only allowed one element per ID (and of course one ID per element).	`#my-id` Selects `<p id="my-id">` or `<a id="my-id">`
Class selector	The element(s) on the page with the specified class (multiple class instances can appear on a page).	`.my-class` Selects `<p class="my-class">` and `<a class="my-class">`
Attribute selector	The element(s) on the page with the specified attribute.	`img[src]` Selects `<img src="myimage.png">` but not `<img>`
Pseudo-class selector	The specified element(s), but only when in the specified state, e.g. being hovered over.	`a:hover` Selects `<a>`, but only when the mouse pointer is hovering over the link.

45

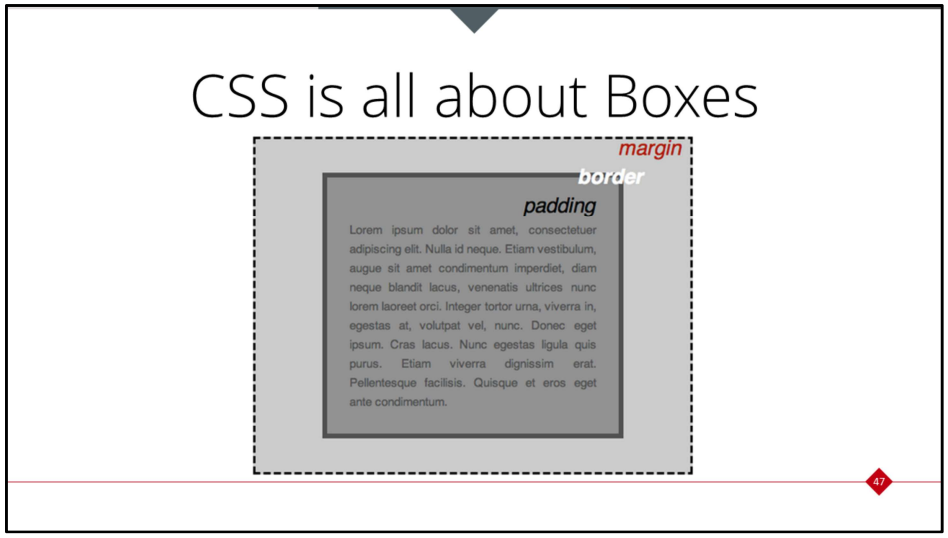
Font and Text

- Create new style.css (delete the red paragraph from before)

```css
html {
  font-size: 10px; /* px means 'pixels': the base font size is
  font-family: 'Open Sans', sans-serif; /* this should be the r
}
```

- We will also stylize header and list

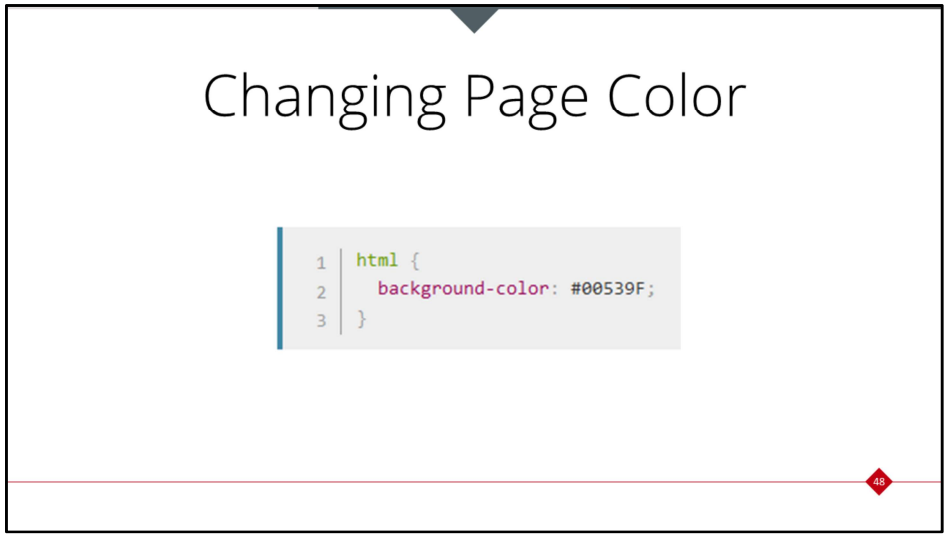
```css
h1 {
  font-size: 60px;
  text-align: center;
}

p, li {
  font-size: 16px;
  line-height: 2;
  letter-spacing: 1px;
}
```



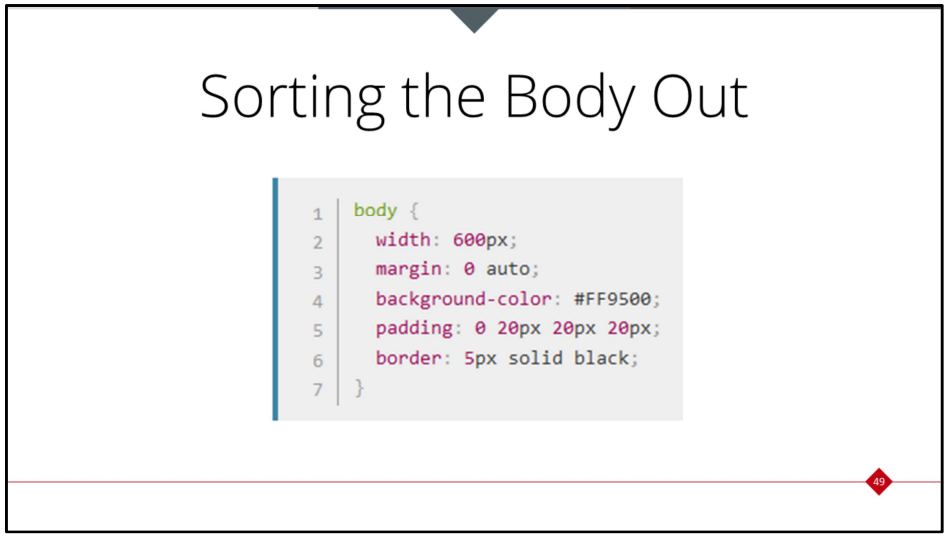
- padding, the space just around the content (e.g., around paragraph text)
- border, the solid line that sits just outside the padding
- margin, the space around the outside of the element

In this section we also use:

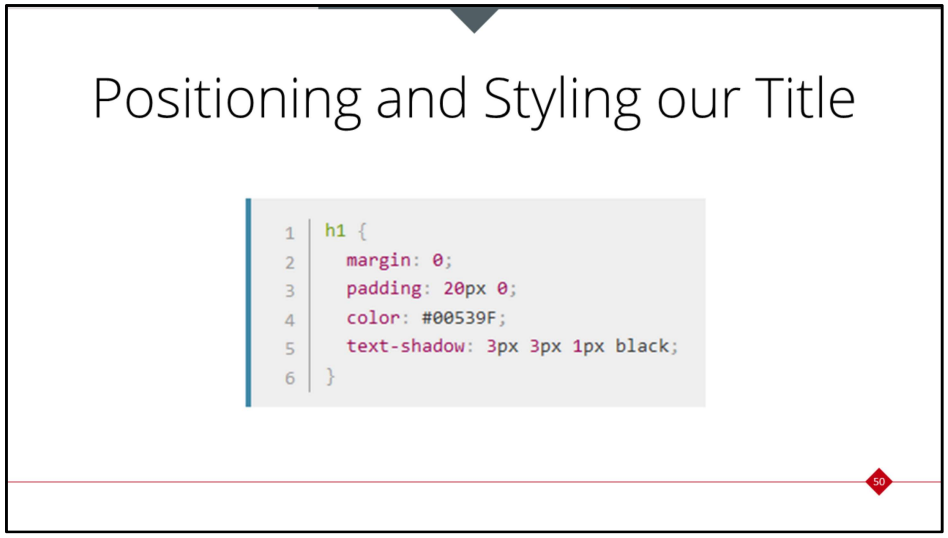
- width (of an element)
- background-color, the color behind an element's content and padding
- color, the color of an element's content (usually text)
- text-shadow: sets a drop shadow on the text inside an element
- display: sets the display mode of an element (don't worry about this yet)



- padding, the space just around the content (e.g., around paragraph text)
- border, the solid line that sits just outside the padding
- margin, the space around the outside of the element



- width: 600px; — this forces the body to always be 600 pixels wide.
- margin: 0 auto; — When you set two values on a property like margin or padding, the first value affects the element's top **and** bottom side (make it 0 in this case), and the second value the left **and** right side (here, auto is a special value that divides the available horizontal space evenly between left and right). You can also use one, three, or four values, as documented here.
- background-color: #FF9500; — as before, this sets the element's background color. We've used a sort of reddish orange for the body as opposed to dark blue for the <html> element, but feel free to go ahead and experiment.
- padding: 0 20px 20px 20px; — we have four values set on the padding, to make a bit of space around our content. This time we are setting no padding on the top of the body, and 20 pixels on the left, bottom and right. The values set top, right, bottom, left, in that order.
- border: 5px solid black; — this simply sets a 5-pixel–wide, solid black border on all sides of the body.

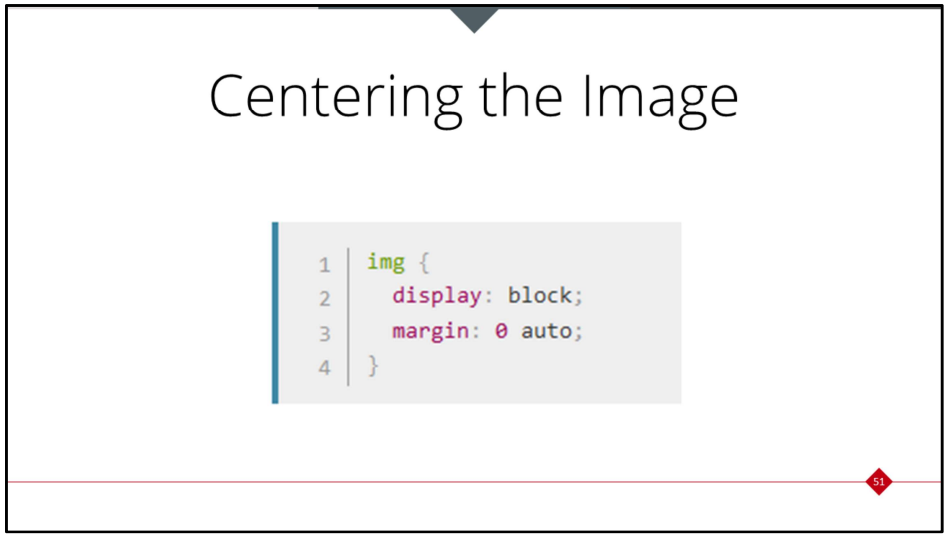


- To get rid of the top gap of H1 we overrode the default styling by setting margin: 0;.

Next up, we've set the heading's top and bottom padding to 20 pixels, and made the heading text the same color as the html background color.

Text Shadow:

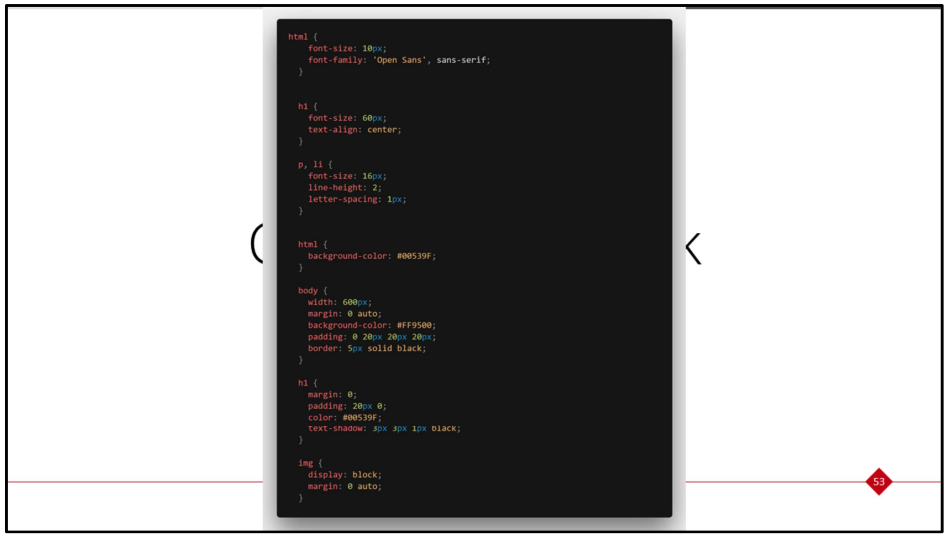
- The first pixel value sets the **horizontal offset** of the shadow from the text — how far it moves across: a negative value should move it to the left.
- The second pixel value sets the **vertical offset** of the shadow from the text — how far it moves down, in this example; a negative value should move it up.
- The third pixel value sets the **blur radius** of the shadow — a bigger value will mean a more blurry shadow.
- The fourth value sets the base color of the shadow.



- We could use the margin: 0 auto trick again as we did earlier for the body, but we also need to do something else. The <body> element is **block level**, meaning it takes up space on the page and can have margin and other spacing values applied to it. Images, on the other hand, are **inline** elements, meaning they can't. So to apply margins to the image, we have to give the image block-level behavior using display: block;.

href = remember that it stands for ***h**ypertext **ref**erence*.

You might get unexpected results if you omit the https:// or http:// part, called the *protocol*, at the beginning of the web address. After making a link, click it to make sure it is sending you where you wanted it to



href = remember that it stands for ***h**ypertext **ref**erence*.

You might get unexpected results if you omit the https:// or http:// part, called the *protocol*, at the beginning of the web address. After making a link, click it to make sure it is sending you where you wanted it to

JavaScript
The Muscle of Our Website

Let's make our website more dynamic and Interactive!

54

What is JavaScript

- JavaScript (JS) is a full-fledged dynamic programming language that, when applied to an HTML document, can provide dynamic interactivity on websites.
- JavaScript is incredibly versatile. You can start small, with carousels, image galleries, fluctuating layouts, and responses to button clicks.
- With more experience, you'll be able to create games, animated 2D and 3D graphics, comprehensive database-driven apps, and much more!

The infamous “Hello World”

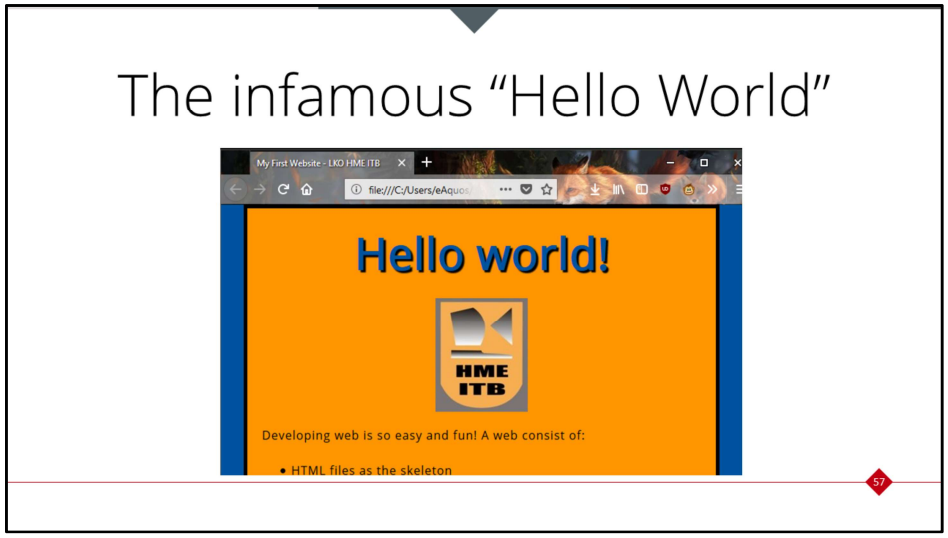
- Create new file `scripts/main.js`
- Inside `index.html` add this line just before </body> tag

```
1 <script src="scripts/main.js"></script>
```

- Now, add this code into `main.js`

```
1 var myHeading = document.querySelector('h1');
2 myHeading.textContent = 'Hello world!';
```

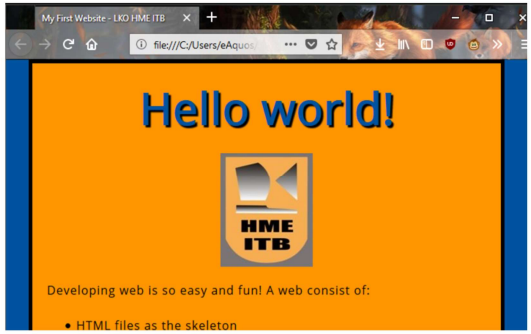
- Then refresh and see what happened!



Nah, bisa dilihat kalo pas direfresh cepat, judulnya berubah. Hal ini nunjukin kalo script pada web bagusnya ditaro di akhir body karena bisa memperlambat loading page dari web.

Selain itu, bisa jadi juga kalau scriptnya ditaro diatas, ga semua elemen kena efek script yang sama. Ga reliable jadinya.

The infamous “Hello World”

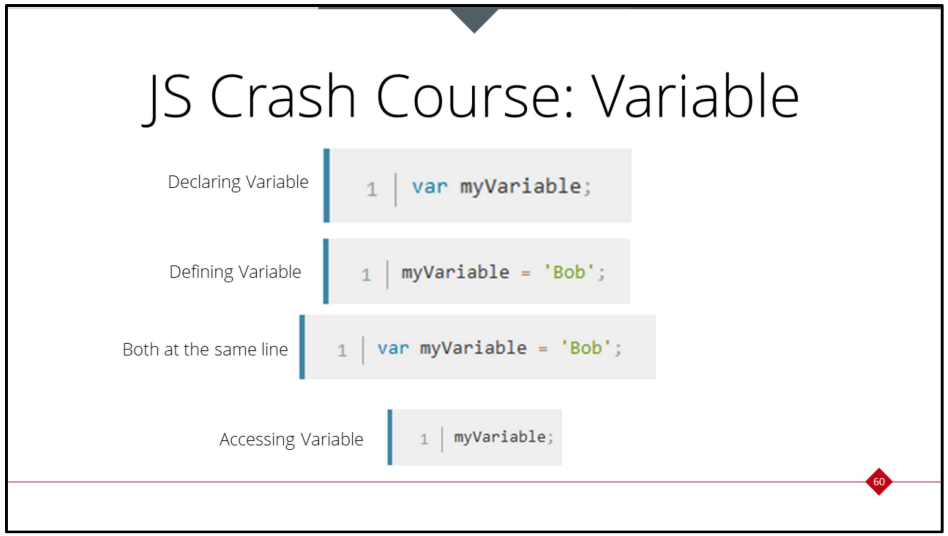


58

You did this by first using a function called querySelector() to grab a reference to your heading, and store it in a variable called myHeading.
This is very similar to what we did using CSS selectors. When wanting to do something to an element, you first need to select it.
After that, you set the value of the myHeading variable's textContent property (which represents the content of the heading) to "Hello world!".

Ini bisa terjadi karena kita tau ada yang namanya Document Object Model. Kita tinggal pake aja.

More about this: https://developer.mozilla.org/en-US/docs/Web/API/Document_Object_Model



A semicolon at the end of a line indicates where a statement ends; it is only absolutely required when you need to separate statements on a single line. However, some people believe that it is a good practice to put them in at the end of each statement. There are other rules for when you should and shouldn't use them — see Your Guide to Semicolons in JavaScript for more details.

You can name a variable nearly anything, but there are some name restrictions (see this article on variable naming rules). If you are unsure, you can check your variable name to see if it is valid.

JavaScript is case sensitive — myVariable is a different variable to myvariable. If you are getting problems in your code, check the casing!

JS Crash Course: Variable

Variable	Explanation	Example
String	A sequence of text known as a string. To signify that the value is a string, you must enclose it in quote marks.	`var myVariable = 'Bob';`
Number	A number. Numbers don't have quotes around them.	`var myVariable = 10;`
Boolean	A True/False value. The words `true` and `false` are special keywords in JS, and don't need quotes.	`var myVariable = true;`
Array	A structure that allows you to store multiple values in one single reference.	`var myVariable = [1,'Bob','Steve',10];` Refer to each member of the array like this: `myVariable[0]`, `myVariable[1]`, etc.
Object	Basically, anything. Everything in JavaScript is an object, and can be stored in a variable. Keep this in mind as you learn.	`var myVariable = document.querySelector('h1');` All of the above examples too.

JS Crash Course: Comment

```
/*
Everything in between is a comment.
*/
```

```
// This is a comment
```

JS Crash Course: Operator

Operator	Explanation	Symbol(s)	Example
Addition	Used to add two numbers together or glue two strings together.	`+`	`6 + 9;` `"Hello " + "world!";`

Subtraction, Multiplication, Division	These do what you'd expect them to do in basic math.	`-`, `*`, `/`	`9 - 3;`
			`8 * 2; // multiply in JS is an asterisk`
			`9 / 3;`

Assignment	You've seen this already: it assigns a value to a variable.	`=`	`var myVariable = 'Bob';`
Equality	Does a test to see if two values are equal to one another and returns a `true/false` (Boolean) result.	`===`	`var myVariable = 3;` `myVariable === 4;`

Not, Does-not-equal	Returns the logically opposite value of what it precedes; it turns a `true` into a `false`, etc. When it is used alongside the Equality operator, the negation operator tests whether two values are *not* equal.	`!`, `!==`	The basic expression is `true`, but the comparison returns `false` because we've negated it:
			`var myVariable = 3;`
			`!(myVariable === 3);`
			Here we are testing "is `myVariable` NOT equal to 3". This returns `false` because `myVariable` IS equal to 3.
			`var myVariable = 3;`
			`myVariable !== 3;`

63

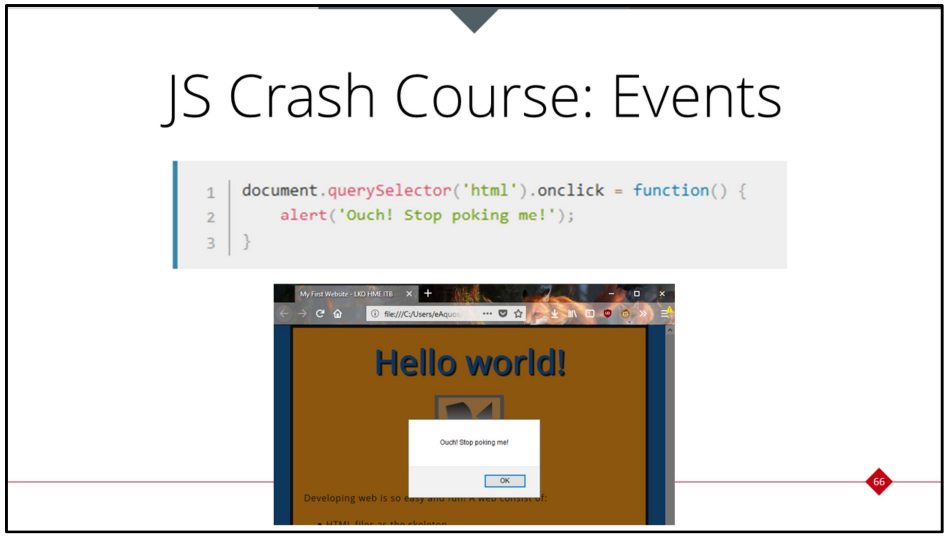
JS Crash Course: Conditional

```
var iceCream = 'chocolate';
if (iceCream === 'chocolate') {
  alert('Yay, I love chocolate ice cream!');
} else {
  alert('Awwww, but chocolate is my favorite...');
}
```

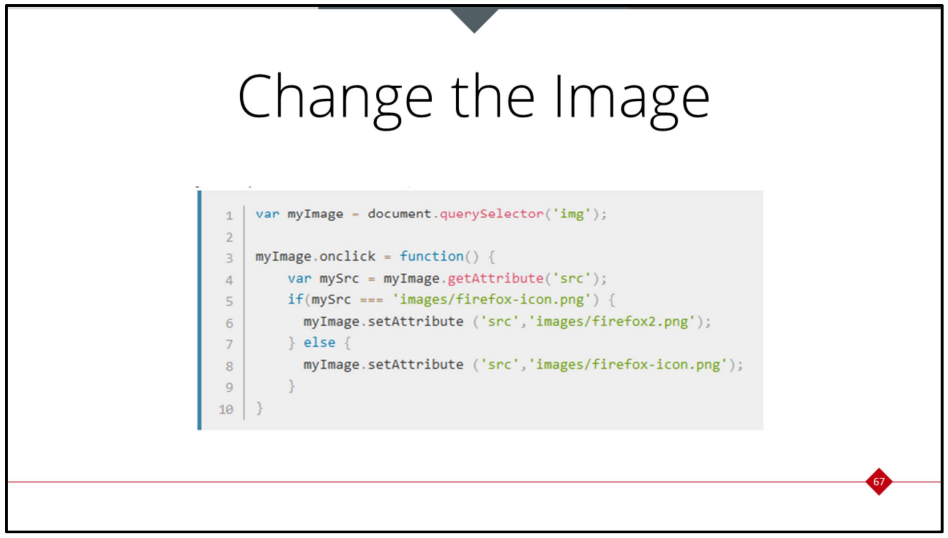
JS Crash Course: Function

```
function multiply(num1,num2) {
  var result = num1 * num2;
  return result;
}
```

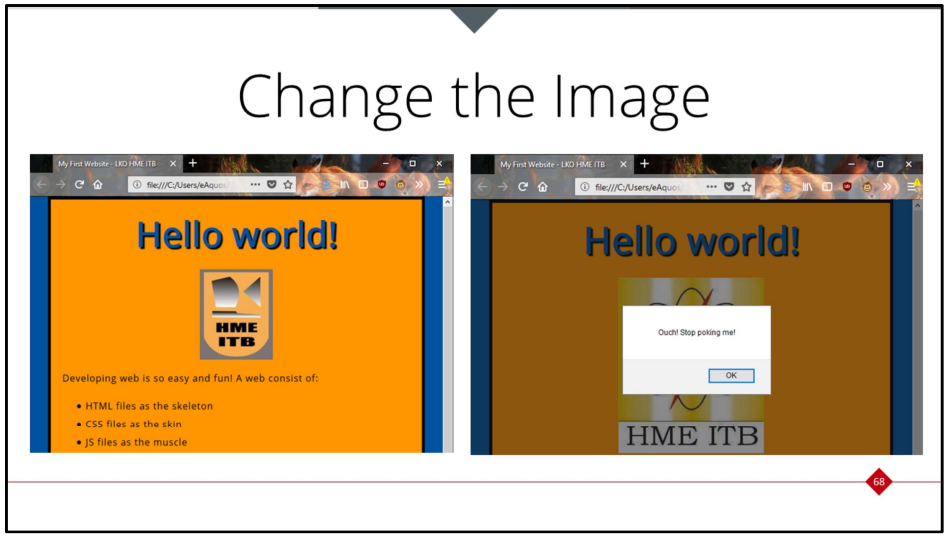
```
multiply(4, 7);
multiply(20, 20);
multiply(0.5, 3);
```



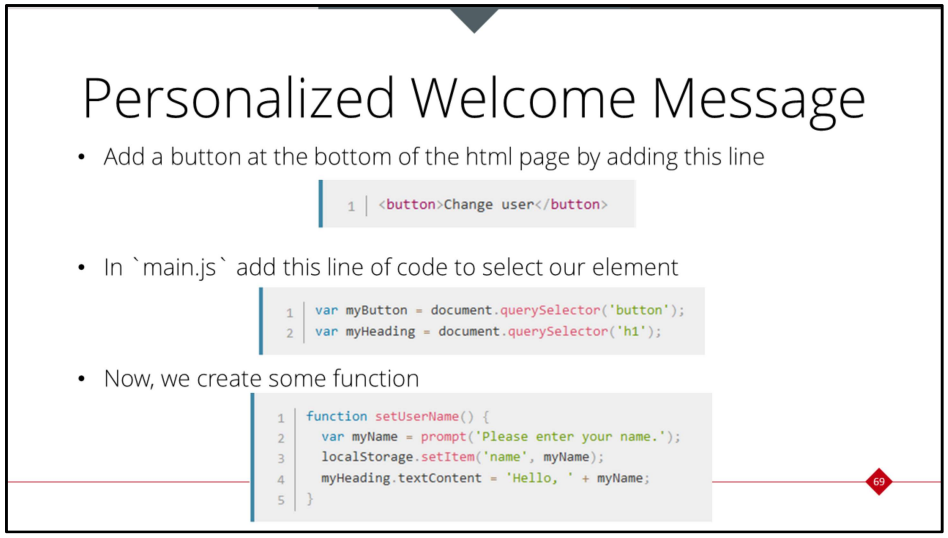
Real interactivity on a website needs events. These are code structures which listen for things happening in browser, running code in response. The most obvious example is the click event, which is fired by the browser when you click on something with your mouse. To demonstrate this, enter the following into your console, then click on the current webpage:



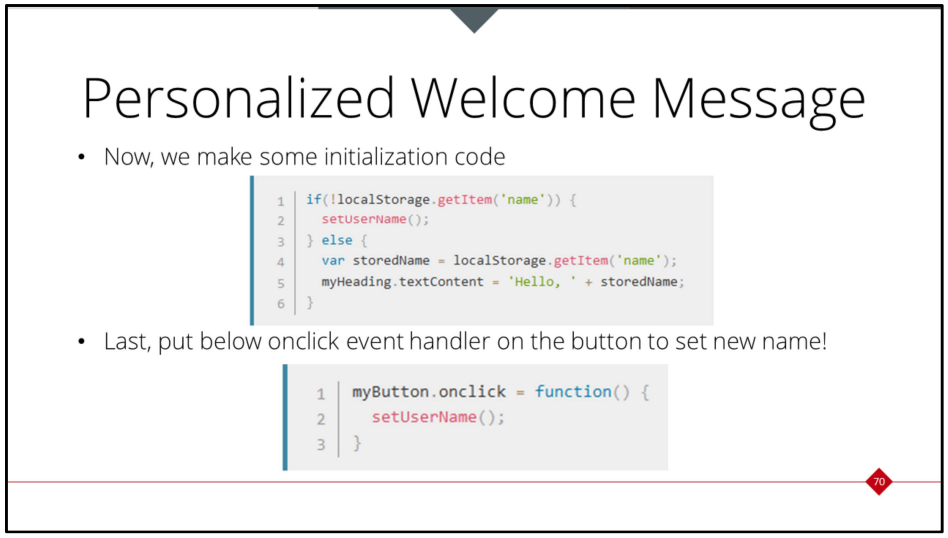
1. You store a reference to your <img> element in the myImage variable. Next, you make this variable's onclick event handler property equal to a function with no name (an "anonymous" function). Now, every time this element is clicked:
2. You retrieve the value of the image's src attribute.
3. You use a conditional to check whether the src value is equal to the path to the original image:
    1. If it is, you change the src value to the path to the 2nd image, forcing the other image to be loaded inside the <img> element.
    2. If it isn't (meaning it must already have changed), the src value swaps back to the original image path, to the original state.



Real interactivity on a website needs events. These are code structures which listen for things happening in browser, running code in response. The most obvious example is the click event, which is fired by the browser when you click on something with your mouse. To demonstrate this, enter the following into your console, then click on the current webpage:

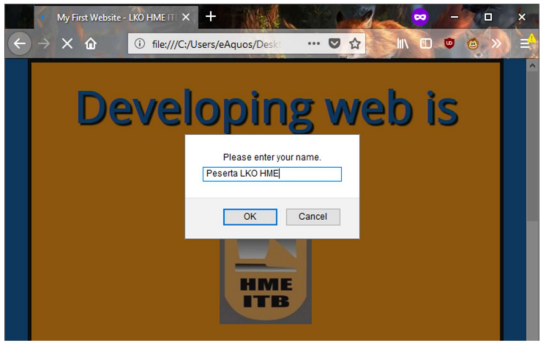


This function contains a prompt() function, which brings up a dialog box, a bit like alert(). This prompt(), however, asks the user to enter some data, storing it in a variable after the user presses **OK**. In this case, we are asking the user to enter their name. Next, we call on an API called localStorage, which allows us to store data in the browser and later retrieve it. We use localStorage's setItem() function to create and store a data item called 'name', setting its value to the myName variable which contains the data the user entered. Finally, we set the textContent of the heading to a string, plus the user's newly stored name.

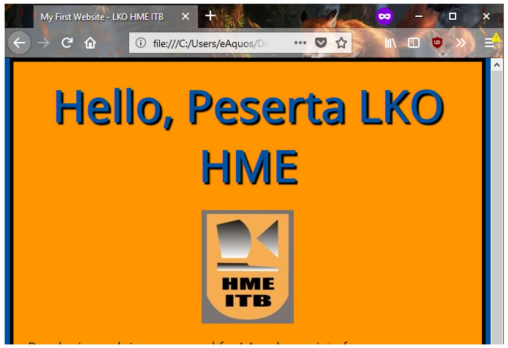


This block first uses the negation operator (logical NOT, represented by the !) to check whether the name data exists. If not, the setUserName() function is run to create it. If it exists (that is, the user set it during a previous visit), we retrieve the stored name using getItem() and set the textContent of the heading to a string, plus the user's name, as we did inside setUserName().

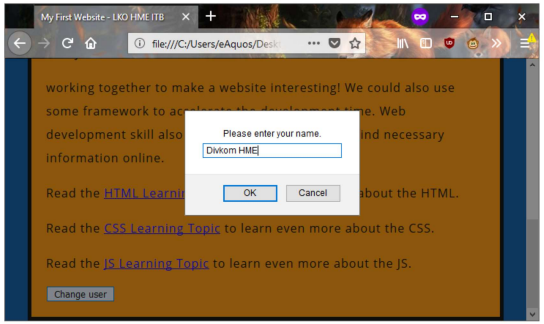
Personalized Welcome Message



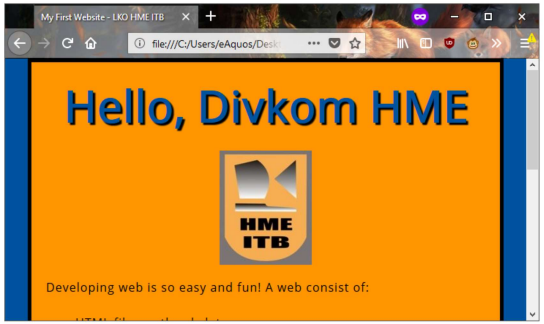
Personalized Welcome Message



Personalized Welcome Message

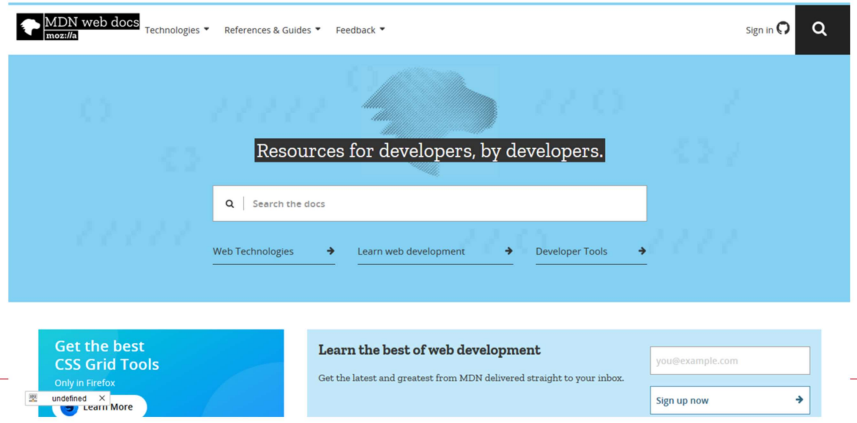


Personalized Welcome Message

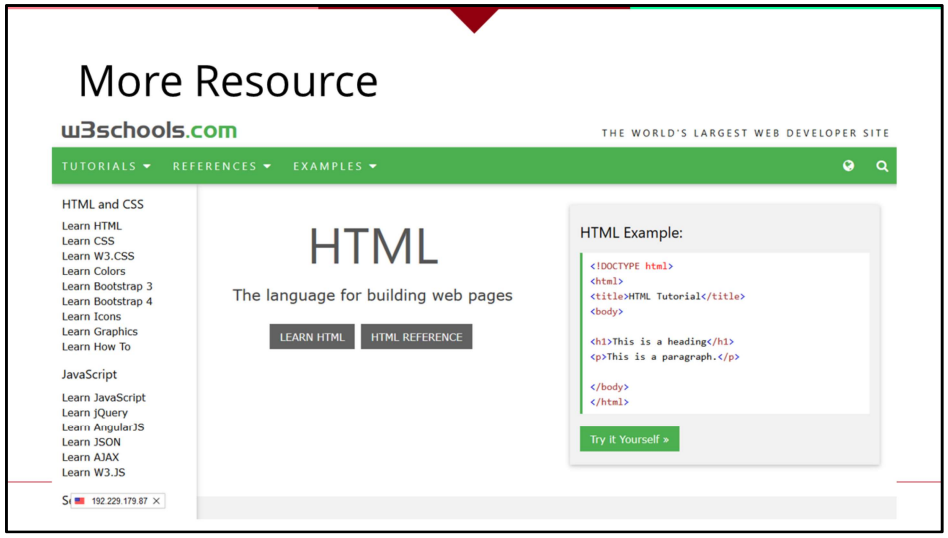


74

More Resource



More Resource



More Resource

More Resource

- Audio Visual Resources:

There and Back Again: A Packet's Tale
How Internet Works in 5 minutes
Web Demystified
Web Development tutorial for Beginners

- Futher Readings (If you like Reading more!):

How does the Internet Works by MDN
Getting started with the Web by MDN
Learn HTML Pathway by MDN
Learn CSS Pathway by MDN
Learn JS Pathway by MDN

The End of Our Journey

Thank you for all your attention! Probably all of this is very comprehensive and tiring to follow. But, worry not because you can review the code later!

This is the introduction to the Front-end Web Development. But, only using this knowledge and a little more effort you can build professional Website in no time! Cool isn't it?

Building website is simple, the most complex part of building a website is solving real problem. Don't let creativity dies inside your head! After all what you need is only text editor, internet connection, and some drinks!