Secure Programming
—— Introduction of the Web

胡天磊, Dr. HU Tianlei

Associate Professor

College of Computer Science, Zhejiang Univ.

htl@zju.edu.cn

Course Outline

- Introduction of HTTP
- Introduction of Front-End Web Language

Hyper Text Transfer Protocol

HTTP - Hyper Text Transfer Protocol

On top of TCP (default port: 80)

Two main versions

Version 1.0 defined in RFC 1945

Version 1.1 defined in RFC 2616

Client

Opens TCP connection to the server

Sends an HTTP request

Server

Accepts the TCP connection from the client

Processes the HTTP request

Sends a HTTP reply

How the HTTP works

Client

Opens TCP connection to the server

Sends an HTTP request

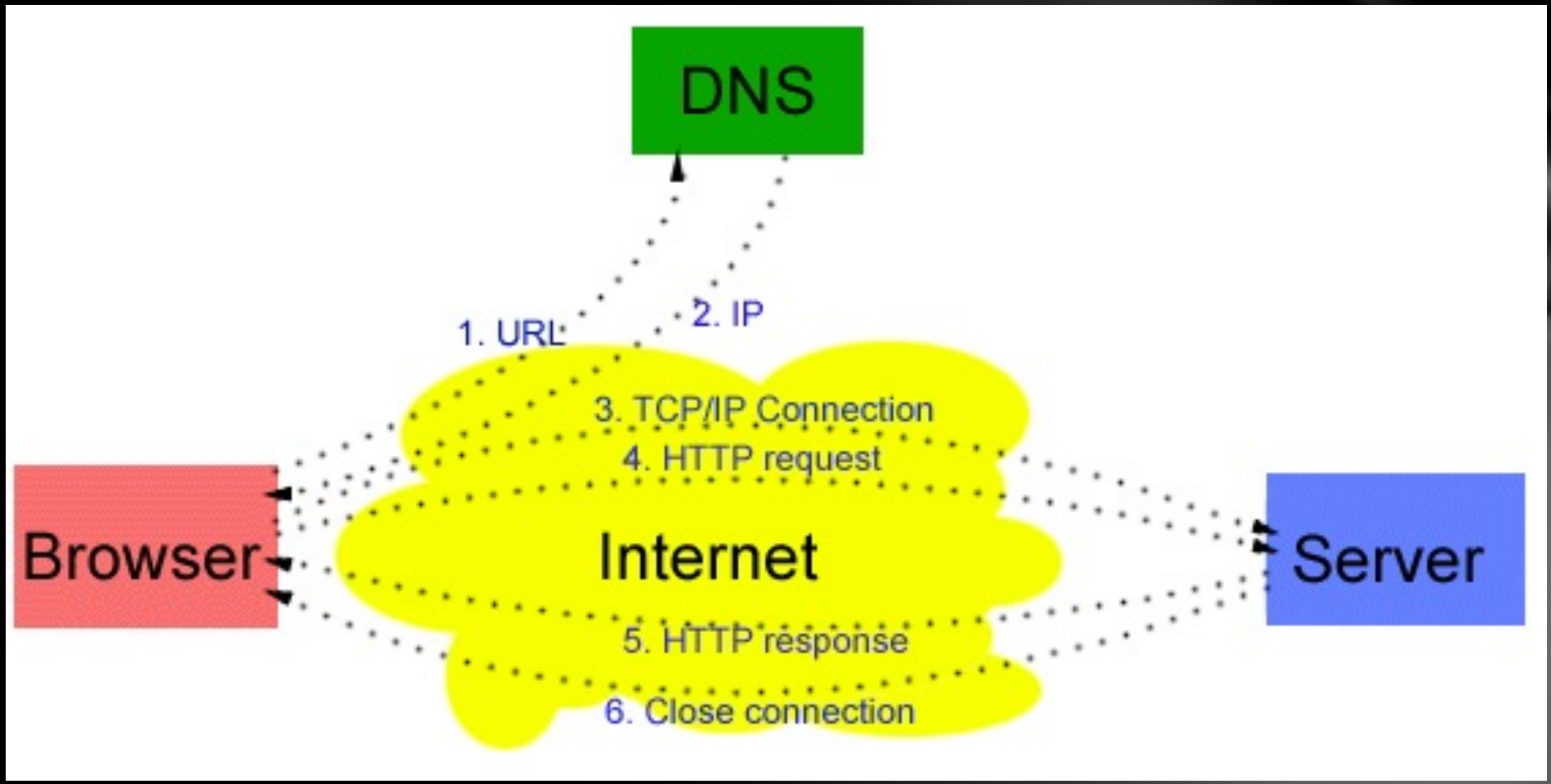
Server

Accepts the TCP connection from the client

Processes the HTTP request

Sends a HTTP reply

How the HTTP works



From one "click" to one Webpage

Short Intro to HTTP

Each request and response has three parts:

- the request or response line
- a header section
- the entity body

The client initiates a transaction as follows:

- GET /index.html?param=value HTTP/1.0

After the request and headers, the client may send additional data

- This data is mostly used by CGI programs using the POST method
- Note that for the GET method, the parameters are encoded into the URL

HTTP Request

An HTTP request consists of the following pieces:

- The method, which must be one of a set of legal actions;
- The Universal Resource Identifier(URI), which is the name of the information requested (Berners-Lee 1994);
- The protocol version;
- (optionally) Other information to modify or supplement the request.

HTTP Request

Example:

```
GET /stuff/Funny/silly.html HTTP/1.0
User-agent: NCSA Mosaic
Accept: text/plain
Accept: text/html
Accept: application/postscript
Accept: image/gif
```

- The first line is the method, the URL, and the version of HTTP(1.0);
- The second line identifies the type of the browser(the User-agent) that is sending the request;
- The following lines indicate the data types (in terms of MIME spec) that the client is prepared to receive.

HTTP Request

Method GET is used in the above example.

In fact, HTTP also has some other methods, which are listed as following:

Method	Explain
GET	Return the object.
HEAD	Return only information about the object, not the object itself.
POST	Send information to be stored on the Server.
PUT	Send a new copy of an existing object to the server.
DELETE	Permanently delete the object.

HTTP Response

An HTTP response consists of the following piece:

- A status line, which indicates the success or failure of the request;
- A description of the information in the response. This is information about the information, which is called meta information;
- Blank line;
- The actual information requested..

HTTP Response

Example:

```
HTTP1.0 Status 200 Document follows
Server: NCSA/1.4
Date: Tue,4 Jul,1996 19:04 GMT
Content-type: text/html
Content-length: 5280
Last-modified: Wed,1 Jan 1996 01:00:02 GMT

The contents of the document requested.
```

- 1st line indicates the version of HTTP that the server uses and the request succeeded and the information will be returned.
- 2nd line indicates the HTTP server software ;
- 3rd line indicates the date and time of the request;
- 4th line indicates this is a HTML document, which is of MIME type text/html;
- 5th line indicates the document is 5280 bytes long;
- 6th line indicates the last modified time of the document;
- Blank line is used as a separating line, indicating the end of meta information.

URLs

Syntax: <scheme>://<authority><path>?<query>

Scheme: a string specifying the protocol/framework

Examples:

- **ftp**://ftp.ietf.org/rfc/rfc1808.txt
- **http**://www.iseclab.org/~dbalzarotti/
- **mailto**:nobody@iseclab.org
- **telnet**://127.0.0.1

URLs

Syntax: <scheme>://<authority><path>?<query>

Authority: a name space that qualifies the resource

- Generally in the form: username@hostname:portnumber
- Hostname can be either a name or an IP address

Examples:

- **ftp://ftp.ietf.org/rfc/rfc1808.txt**
- **http://www.iseclab.org/~dbalzarotti/**
- **mailto:nobody@iseclab.org**
- **telnet://127.0.0.1**

URLs

Syntax: <scheme>://<authority><path>?<query>

Path: a / separated path of the requested resource

Examples:

- ftp://ftp.ietf.org/rfc/rfc1808.txt
- http://www.iseclab.org/~dbalzarotti/
- mailto:nobody@iseclab.org
- telnet://127.0.0.1

URLs

Syntax: <scheme>://<authority><path>?<query>

Query: application-specific piece of information

Examples:

- **http://host/login.php?user=foo&pwd=bar**

HTTPs

Encrypt the communication

- Protect against eavesdropping
- Protect from man in the middle attacks
  (provided that the certificate is trusted)
- Used to protect the user authentication
- By-pass IDS / IPS

The trust inherent in HTTPS is based on major certificate authorities that come pre-installed in browser software.

- Does not protect from attacks against the web application.

Web Languages
——HTML/CSS/JS

Web Languages

3 languages all web developers MUST learn:

- **HTML to define the content of web pages**
- **CSS to specify the layout of web pages**
- **JavaScript to specify the behavior of web pages**

What is HTML?

HTML is a language for describing web pages.

- HTML stands for Hyper Text Markup Language
- HTML is a markup language
- A markup language is a set of markup tags
- The tags describe document content
- HTML documents contain HTML tags and plain text
- HTML documents are also called web pages

HTML Example

```
<!DOCTYPE html>
< html>
< body>

< h1>My First Heading</h1>

< p>My first paragraph.</p>

< /body>
< /html>
```

Example Explained

- The DOCTYPE declaration defines the document type
- The text between <html> and </html> describes the web page
- The text between <body> and </body> is the visible page content
- The text between <h1> and </h1> is displayed as a heading
- The text between and is displayed as a paragraph

HTML Tags

HTML markup tags are usually called HTML tags.

- HTML tags are keywords (tag names) surrounded by angle brackets like <html>
- HTML tags normally come in pairs like and
- The first tag in a pair is the start tag, the second tag is the end tag
- The end tag is written like the start tag, with a forward slash before the tag name
- Start and end tags are also called opening tags and closing tags

<tagname>content</tagname>

HTML Elements

"HTML tags" and "HTML elements" are often used to describe the same thing.

- But strictly speaking, an HTML element is everything between the start tag and the end tag, including the tags.

This is a paragraph.

HTML Headings

Headings are defined with the <h1> to <h6> tags. <h1> defines the most important heading. <h6> defines the least important heading.

```
<h1>This is a heading.</h1>
<h2>This is a heading.</h2>
<h3>This is a heading.</h3>
```

Headings Are Important

- Use HTML headings for headings only. Don't use headings to make text BIG or bold.
- Search engines use your headings to index the structure and content of your web pages.
- Since users may skim your pages by its headings, it is important to use headings to show the document structure.
- H1 headings should be used as main headings, followed by H2 headings, then the less important H3 headings, and so on.

HTML Paragraphs

Paragraphs are defined with the <p> tag.

```
<p>This is a paragraph.</p>
<p>This is another paheading.</p>
```

Use the
 tag if you want a line break (a new line) without starting a new paragraph

```
< p>This is<br>a para<br>graph with line
breaks.</p>
```

HTML Text Formatting

This text is bold

This text is italic

This is _{subscript} and ^{superscript}

HTML Text Formatting

Tag	Description
<b>	Defines bold text
	Defines emphasized text
<i>	Defines a part of text in an alternate voice or mood
<small>	Defines smaller text
	Defines important text
<sub>	Defines subscripted text
<sup>	Defines superscripted text
<ins>	Defines inserted text
<del>	Defines deleted text
<mark>	Defines marked/highlighted text

HTML Comments

Comment tags <!-- and --> are used to insert comments in HTML.

<!-- Write your comments here -->

Comments are not displayed by the browser, but they can help document your HTML. With comments you can place notifications and reminders in your HTML.

HTML Links

The HTML <a> tag defines a hyperlink.

- A hyperlink (or link) is a word, group of words, or image that you can click on to jump to another document.
- When you move the cursor over a link in a Web page, the arrow will turn into a little hand.
- The most important attribute of the <a> element is the href attribute, which indicates the link's destination.

```
<a href="url">Link text</a>
```

HTML Head

The <head> element is a container for all the head elements.

- The following tags can be added to the head section: <title>, <style>, <meta>, <link>, <script>, <noscript>, and <base>.

```
<head>
  < title>Title of the document</title>
  <base href="http://www.w3schools.com/images/"
    target="_blank">
  <link rel="stylesheet" type="text/css" href="mystyle.css">
  <style type="text/css">
     body {background-color:yellow;}
     p {color:blue;}
   </style>
< /head>
```

HTML Head Related Tags

Tag	Description
<head>	Defines information about the document
<title>	Defines the title of a document
<base>	Defines a default address or a default target for all links on a page
<link>	Defines the relationship between a document and an external resource
<meta>	Defines metadata about an HTML document
<script>	Defines a client-side script
<style>	Defines style information for a document

HTML Tag Reference

More Tags

There are also other tags, like CSS, Images, Tables, Lists, Forms, Blocks, Layout, and so on.

More Info

CSS Tutorial: http://www.w3schools.com/css/default.asp

HTML DOM

What's DOM

DOM is a W3C (World Wide Web Consortium) standard.

- Short for "Document Object Model".
- "The W3C Document Object Model (DOM) is a platform and language-neutral interface that allows programs and scripts to dynamically access and update the content, structure, and style of a document."

The W3C DOM standard is separated into 3 different parts:

- Core DOM - standard model for all document types
- XML DOM - standard model for XML documents
- HTML DOM - standard model for HTML documents

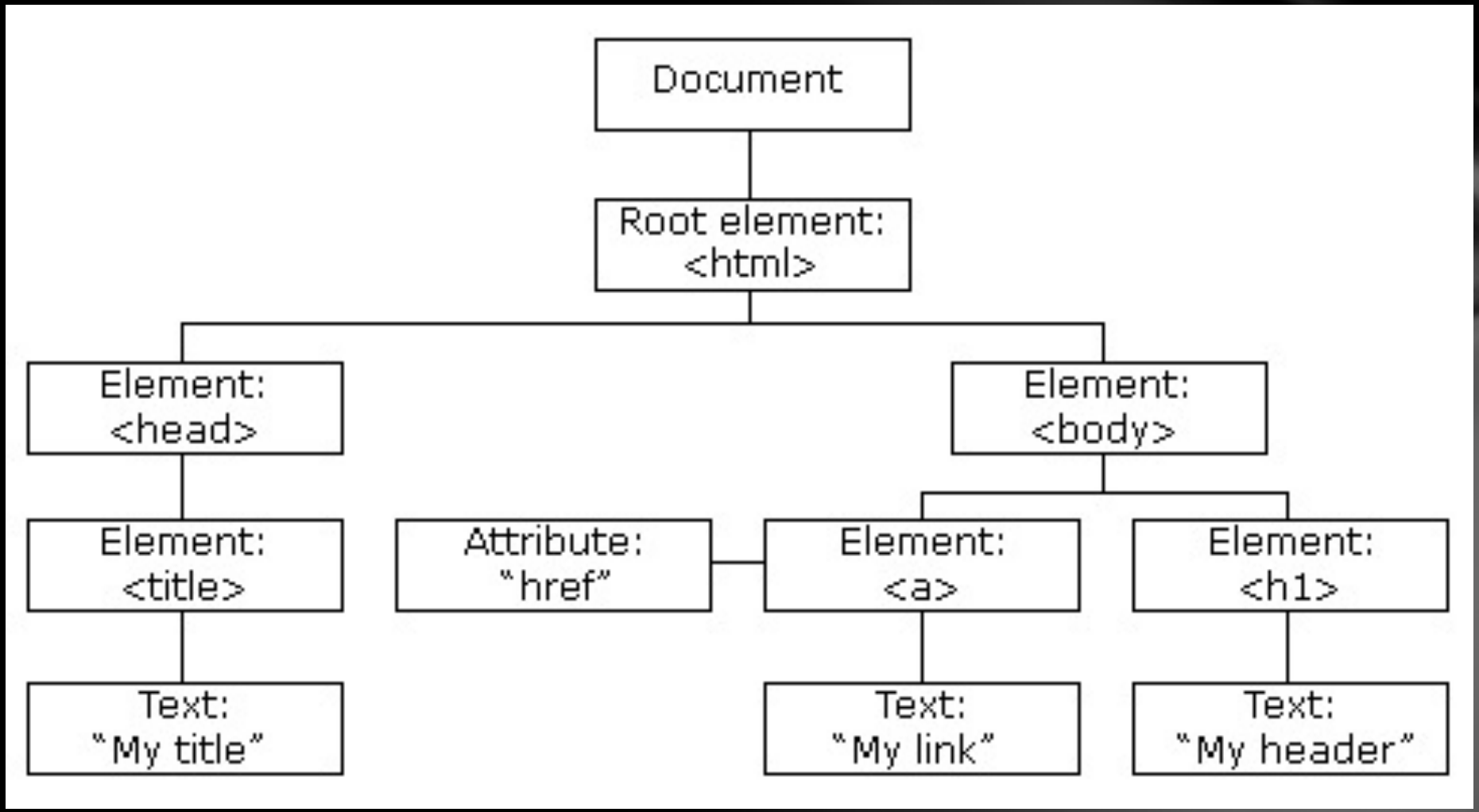
What's HTML DOM

The HTML DOM is a standard object model and programming interface for HTML. It defines:

- The HTML elements as objects
- The properties of all HTML elements
- The methods to access all HTML elements
- The events for all HTML elements

In other words: The HTML DOM is a standard for how to get, change, add, or delete HTML elements.

HTML DOM


Document
Root element:
<html>
Element:
<head>
Element:
<body>
Element:
<title>
Attribute:
"href"
Element:
<a>
Element:
<h1>
Text:
"My title"
Text:
"My link"
Text:
"My header"

Web Languages

——CSS

CSS

What's CSS

- CSS stands for Cascading Style Sheets
- Styles define how to display HTML elements

CSS can be added to HTML in the following ways:

- Inline - using the style **attribute** in HTML elements
- Internal - using the <style> **element** in the <head> section
- External - using an external CSS **file**

The preferred way to add CSS to HTML, is to:

put CSS syntax in separate CSS files.

CSS Solves a Big Problem

HTML was never intended to contain tags for formatting a document.

- HTML was intended to define the content of a document, like:

```
<h1>This is a heading</h1>
<p>This is a paragraph.</p>
```

- When tags like <font>, and color attributes were added to the HTML 3.2 specification, it started a nightmare for web developers.
  - Development of large web sites, where fonts and color information were added to every single page, became a long and expensive process.
  - To solve this problem, the World Wide Web Consortium (W3C) created CSS.
- In HTML 4.0, all formatting could be removed from the HTML document, and stored in a separate CSS file.

All browsers support CSS today.

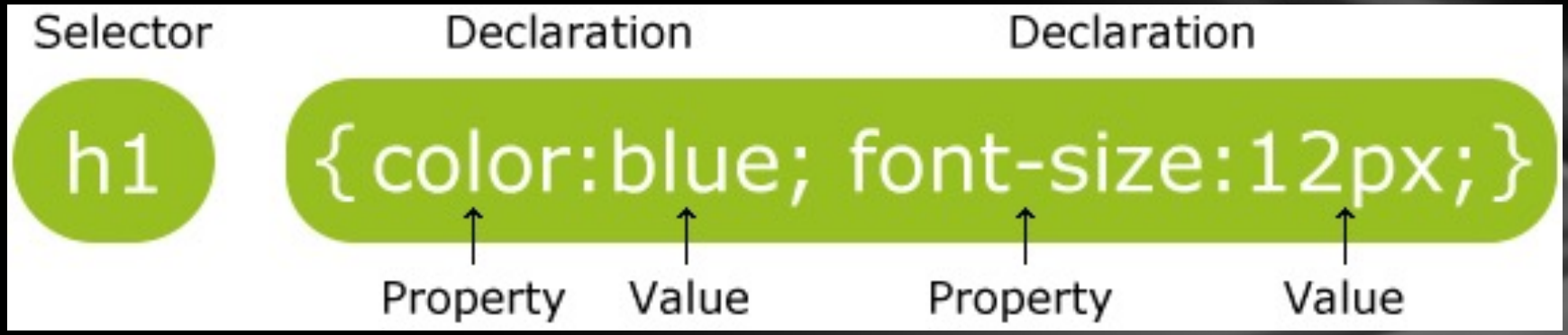
CSS Saves a Lot of Work

CSS defines HOW HTML elements are to be displayed.

- Styles are normally saved in external .css files.
- External style sheets enable you to change the appearance and layout of all the pages in a Web site, just by editing one single file!

CSS Syntax

- A CSS rule set consists of a selector and a declaration block:



- The selector points to the HTML element you want to style.
- The declaration block contains one or more declarations separated by semicolons.
- Each declaration includes a property name and a value, separated by a colon.

CSS Example

- A CSS declaration always ends with a semicolon, and declaration groups are surrounded by curly brackets:

```
p {color: red; text-align: center;}
```

- To make the CSS more readable, you can put one declaration on each line, like this:

```
p
{
color: red;
text-align: center;
}
```

CSS Example

- A CSS comment starts with /* and ends with */:

```
/*This is a multiple
lines comment*/
p
{
color: red;
/*This is another comment*/
text-align: center;
}
```

How To …

When a browser reads a style sheet, it will format the document according to it.

Three Ways to Insert CSS

- External style sheet
- Internal style sheet
- Inline style

External Style Sheet

An external style sheet is ideal when the style is applied to many pages. With an external style sheet, you can change the look of an entire Web site by changing one file. Each page must link to the style sheet using the <link> tag. The <link> tag goes inside the head section:

```
<head>
<link rel="stylesheet" type="text/css"
href="mystyle.css">
</head>
```

```
hr {color: sienna;}
p {margin-left:20px;}
body {  background-image:
        url("images/background.gif");}
```

Internal Style Sheet

An internal style sheet should be used when a single document has a unique style. You define internal styles in the head section of an HTML page, by using the <style> tag, like this:

```
<head>
<style>
hr {color: sienna;}
p {margin-left:20px;}
body {background-image:
        url("images/background.gif");}
</style>
</head>
```

Inline Styles

An inline style loses many of the advantages of style sheets by mixing content with presentation. Use this method sparingly!

- To use inline styles you use the style attribute in the relevant tag.
- The style attribute can contain any CSS property.
- The example shows how to change the color and the left margin of a paragraph:

```
<p style="color:sienna;margin-left:20px;">This is a paragraph.</p>
```

Multiple Style Sheets

If some properties have been set for the same selector in different style sheets, the values will be inherited from the more specific style sheet.

For example, an external style sheet has these properties for the h3 selector:

```
h3
{
   color: red;
   text-align: left;
   font-size:8pt;
}
```

Multiple Style Sheets

And an internal style sheet has these properties for the h3 selector:

```
h3
{
   text-align: right;
   font-size: 20pt;
}
```

If the page with the internal style sheet also links to the external style sheet the properties for h3 will be:

```
color: red;
text-align: right;
font-size: 20pt;
```

The color is inherited from the external style sheet and the text-alignment and the font-size is replaced by the internal style sheet.

Cascading order

What style will be used when there is more than one style specified for an HTML element?

Generally speaking we can say that all the styles will "cascade" into a new "virtual" style sheet by the following rules, where number 4 has the highest priority:

1. ***Browser default***
2. ***External style sheet***
3. ***Internal style sheet (in the head section)***
4. ***Inline style (inside an HTML element)***

Web Languages
——JavaScript

JavaScript

- Developed by Netscape as a light-weight scripting language with object-oriented capabilities
  - Current version standardized as ECMA 357
  - Most popular scripting language on the Internet
  - Works with basically all browsers
- Designed to add interactivity to HTML pages
  - usually embedded directly into HTML pages (<script>tags)
  - can access and add elements to HTML page (DOM tree)
  - can react to events
- JavaScript is a scripting language
  - dynamic, weak typing
  - interpreted language
  - script executes on virtual machine in browser

JavaScript

- JavaScript is quite different from Java
  - Originally, JavaScript was named LiveScript
    - marketing department made developers change the name
  - Java is more complex and powerful
  - static, strong typing

- Design decisions (Brendan Eich)
  - make it easy to copy and paste snippets of code
  - tolerate "minor" errors (missing semicolons)
  - simplified event handling
  - choose some powerful, often-needed primitives

JavaScript

- Syntax quite similar to Java
  - control statements, exception handling
- No classes, but object-based
  - uses objects with properties (name - value pairs)
- No input / output facilities per se
  - must be provided by embedding environment
- Scope of variables is either global or function-local
- Code can be generated at run-time and executed on-the-fly
  - eval() function

Where To Write Code

In HTML, JavaScripts must be inserted between <script> and </script> tags where can be put in the <body> and in the <head> section of an HTML page.

Example

```
<script>
function myFunction()
{
   document.getElementById("demo").innerHTML="My First
JavaScript Function";
}
</script>
```

The browser will interpret the code between the <script> and </script> tags as JavaScript.

JavaScript Functions and Events

- Most often, JavaScript code is written to be executed when an event occurs, like when the user clicks a button.
- If we put JavaScript code inside a function, we can call that function when an event occurs.

JavaScript in <head> or <body>

- You can place an unlimited number of scripts in an HTML document.
- Scripts can be in the <body> or in the <head> section of HTML, and/or in both.
- It is a common practice to put functions in the <head> section, or at the bottom of the page.
- Separating HTML and JavaScript, by putting all the code in one place, is always a good habit.

JavaScript in <head>

Example

```
<!DOCTYPE html>
<html><head>
<script>
function myFunction()
{
document.getElementById("demo").innerHTML="My First
JavaScript Function";
}
</script>
</head>
<body>
<h1>My Web Page</h1>
<p id="demo">A Paragraph</p>
<button type="button" onclick="myFunction()">Try it</button>
</body>
</html>
```

JavaScript in <body>

Example

```
<!DOCTYPE html>
<html>
<body>
<h1>My Web Page</h1>
<p id="demo">A Paragraph</p>
<button type="button" onclick="myFunction()">Try it</button>

<script>
function myFunction()
{
document.getElementById("demo").innerHTML="My First
JavaScript Function";
}
</script>

</body>
</html>
```

External JavaScript

Scripts can also be placed in external files. External files often contain code to be used by several different web pages.

- External JavaScript files have the file extension .js.
- To use an external script, put the name of the script file in the source (src) attribute of the <script> tag.

```
<!DOCTYPE html>
<html>
<body>
<script src="myScript.js"></script>
</body>
</html>
```

JavaScript Output

- JavaScript does not have any print or output functions.
- In HTML, JavaScript can only be used to manipulate HTML elements.
- To use

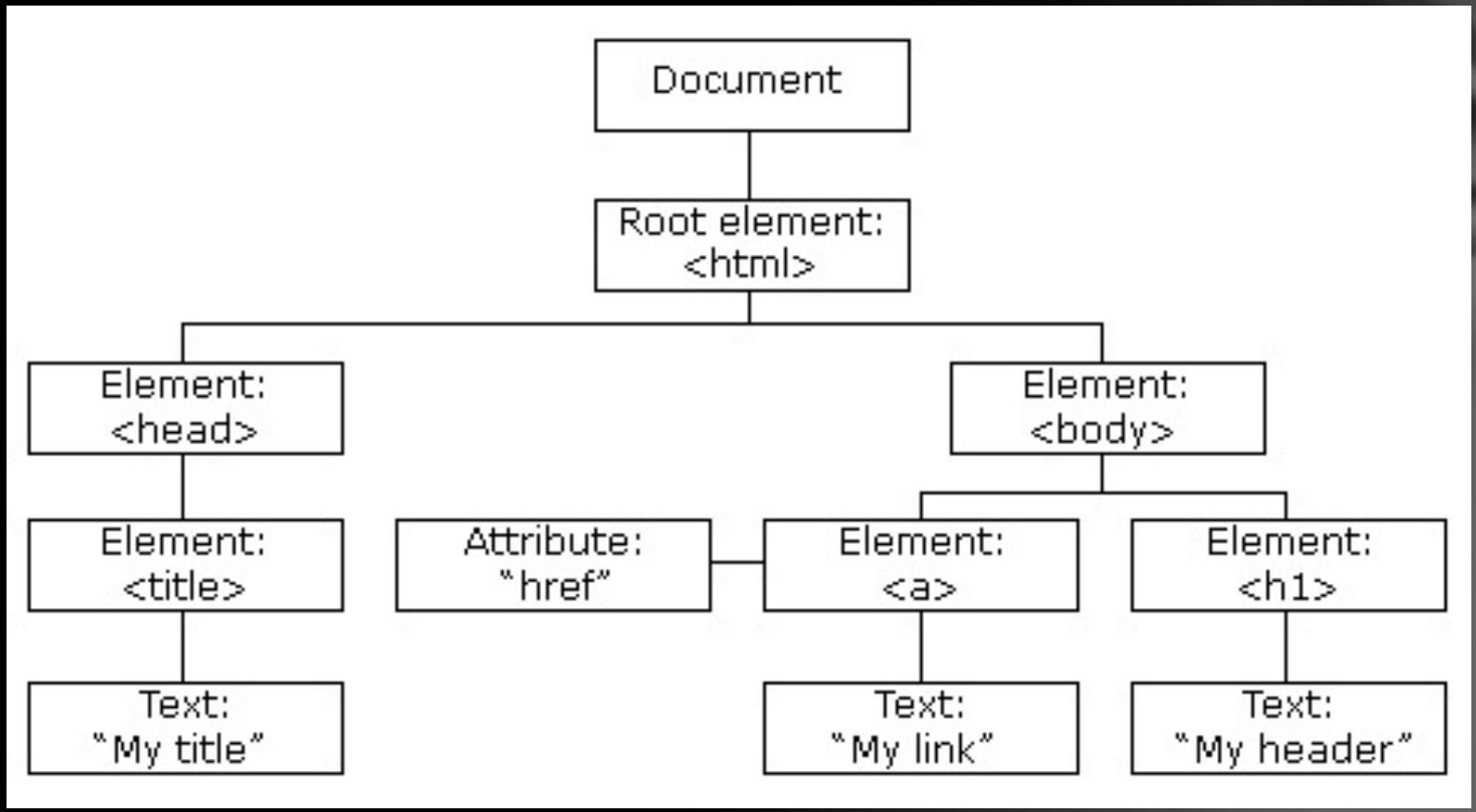
```
<!DOCTYPE html>
<html>
<body>
<h1>My First Web Page</h1>
<p id="demo">My First Paragraph</p>

<script>
elem = document.getElementById("demo");
elem.innerHTML = "My First JavaScript";
</script>

</body>
</html>
```

src)
attrib

Mani

HTML DOM

When a web page is loaded, the browser creates a Document Object Model of the page. The HTML DOM model is constructed as a tree of Objects:



What the JavaScript can DO

JavaScript can change all the HTML elements in the page

- JavaScript can change all the HTML attributes in the page
- JavaScript can change all the CSS styles in the page
- JavaScript can remove existing HTML elements and attributes
- JavaScript can add new HTML elements and attributes
- JavaScript can react to all existing HTML events in the page
- JavaScript can create new HTML events in the page

The DOM Programming Interface

The HTML DOM can be accessed with JavaScript (and with other programming languages).

- In the DOM, all HTML elements are defined as objects.
- The programming interface is the properties and methods of each object.
  - A property is a value that you can get or set (like changing the content of an HTML element).
  - A method is an action you can do (like add or deleting an HTML element).

Example

```
<html>
<body>
<p id="intro">Hello World!</p>

<script>
   var txt=document.getElementById("intro").innerHTML;
   document.write(txt);
</script>

</body>
</html>
```

In the example above, *getElementById()* is a method, while *innerHTML* is a property.

The DOM Programming Interface

The getElementById() method

- The most common way to access an HTML element is to use the id of the element.
- In the example above the getElementById() method used id="intro" to find the element.

The innerHTML property

- The easiest way to get the content of an element is by using the innerHTML property.
- The innerHTML property is useful for getting or replacing the content of HTML elements.

The Methods

Finding HTML Elements

Method	Description
document.getElementById()	Finding an element by element id
document.getElementsByTagName()	Finding elements by tag name
document.getElementsByClassName()	Finding elements by class name
document.forms[]	Finding elements by HTML element objects

The Methods

Changing HTML Elements

Method	Description
element.innerHTML=	Changing the inner HTML of an element
element.*attribute*=	Changing the attribute of an HTML element
element.setAttribute(*attribute,value*)	Changing the attribute of an HTML element
element.style.*property*=	Changing the style of an HTML element

The Methods

Adding and Deleting Elements

Method	Description
document.createElement()	Create an HTML element
document.removeChild()	Remove an HTML element
document.appendChild()	Add an HTML element
document.replaceChild()	replace an HTML element
document.write(*text*)	Writing into the HTML output stream

Adding Events Handlers

Method	Description
document.getElementById(*id*).onclick=function(){*code*}	Adding event handler code to an onclick event

HTML DOM Navigation

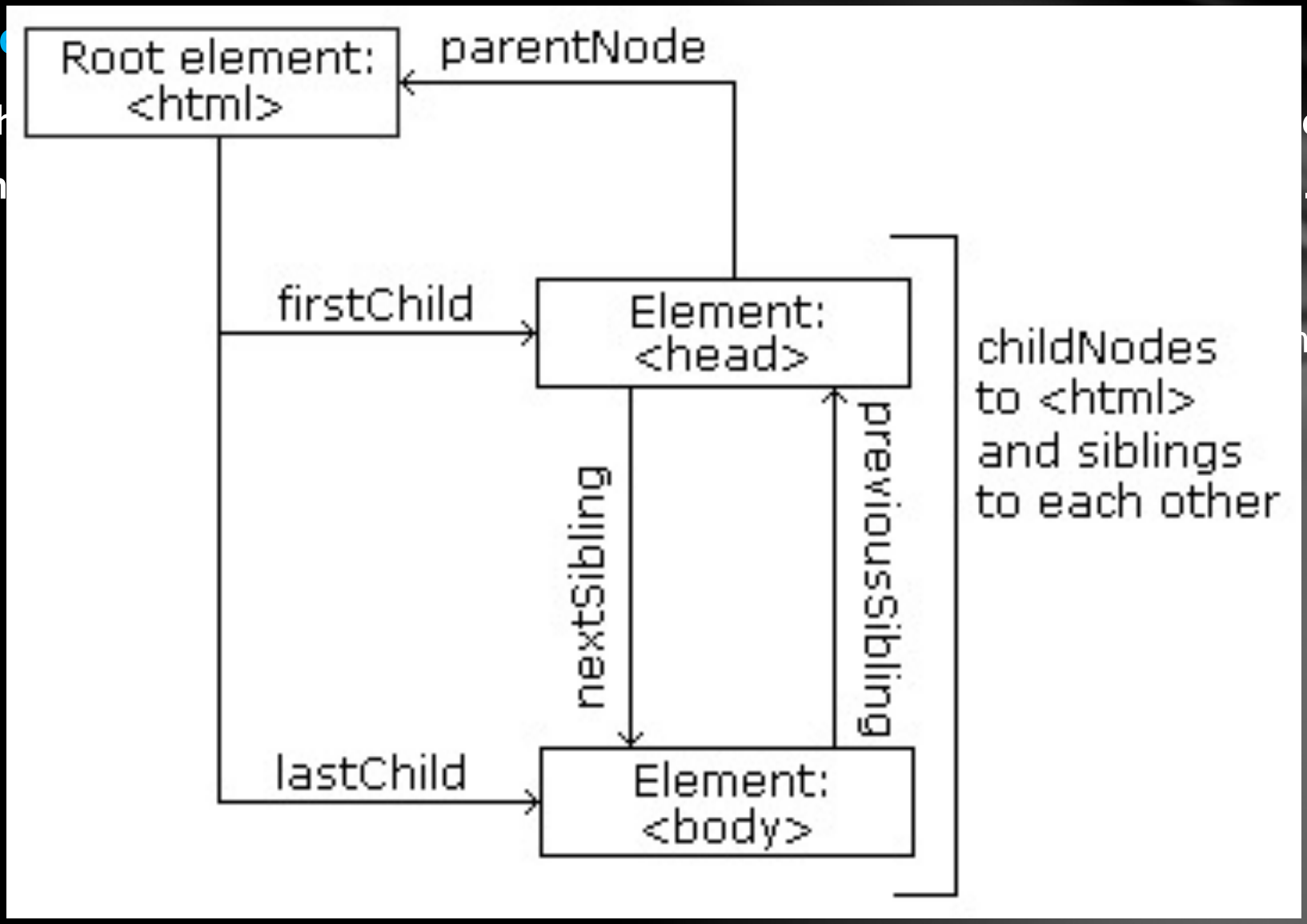
DOM Nodes

According to W3C HTML DOM standard, everything in an HTML document is a node:

- The entire document is a document node
- Every HTML element is an element node
- The text inside HTML elements are text nodes
- Every HTML attribute is an attribute node
- All comments are comment nodes

With the HTML DOM, all nodes in the node tree can be accessed by JavaScript. New nodes can be created, and all nodes can be modified or deleted.

HTML DOM Navigation



HTML DOM Navigation

Example

```
<html>

 <head>
  <title>DOM Tutorial</title>
 </head>

 <body>
  <h1>DOM Lesson one</h1>
  <p>Hello world!</p>
 </body>

</html>
```

HTML DOM Navigation

Example

From the HTML above you can read:

- <html> is the root node
- <html> has no parents
- <html> is the parent of <head> and <body>
- <head> is the first child of <html>
- <body> is the last child of <html>

and:

- <head> has one child: <title>
- <title> has one child (a text node): "DOM Tutorial"
- <body> has two children: <h1> and <p>
- <h1> has one child: "DOM Lesson one"
- <p> has one child: "Hello world!"
- <h1> and <p> are siblings

HTML DOM Navigation

Navigating Between Nodes

You can use the following node properties to navigate between nodes with JavaScript:

- parentNode
- childNodes[nodenumber]
- firstChild
- lastChild
- nextSibling
- previousSibling

HTML DOM Navigation

Child Nodes and Node Values

In addition to the innerHTML property, you can also use the childNodes and nodeValue properties to get the content of an element.

The following code gets the value of the <p> element with id="intro":

```
<html>
<body>
<p id="intro">Hello World!</p>
<script>
var
txt=document.getElementById("intro").childNodes[0].node
Value;
document.write(txt);
</script>
</body>
</html>
```

HTML DOM Navigation

DOM Root Nodes

There are two special properties that allow access to full document:

- document.documentElement - The full document
- document.body - The body of the document

```
<html>
<body>
<p>Hello World!</p>
<div>
<p>The DOM is very useful!</p>
<p>This example demonstrates the <b>document.body</b> property.</p>
</div>
<script>
alert(document.body.innerHTML);
</script>
</body>
</html>
```

HTML DOM Navigation

The nodeName Property

The nodeName property specifies the name of a node.

- nodeName is read-only
- nodeName of an element node is the same as the tag name
- nodeName of an attribute node is the attribute name
- nodeName of a text node is always #text
- nodeName of the document node is always #document

Note: nodeName always contains the uppercase tag name of an HTML element.

HTML DOM Navigation

The nodeValue Property

The nodeValue property specifies the value of a node.

- nodeValue for element nodes is undefined
- nodeValue for text nodes is the text itself
- nodeValue for attribute nodes is the attribute value

HTML DOM Navigation

The nodeType Property

The nodeType property returns the type of node. nodeType is read only. The most important node types are:

Element type	NodeType
Element	1
Attribute	2
Text	3
Comment	8
Document	9

HTML DOM Elements (Nodes)

Creating New HTML Elements (Nodes)

To add a new element to the HTML DOM, you must create the element (element node) first, and then append it to an existing element.

```
<div id="div1">
<p id="p1">This is a paragraph.</p>
<p id="p2">This is another paragraph.</p>
</div>

<script>
var para=document.createElement("p");
var node=document.createTextNode("This is new.");
para.appendChild(node);
var element=document.getElementById("div1");
element.appendChild(para);
</script>
```

HTML DOM Elements (Nodes)

Creating new HTML Elements - insertBefore()

The appendChild() method in the previous example, appended the new element as the last child of the parent.

If you don't want that you can use the insertBefore() method:

```
<div id="div1">
<p id="p1">This is a paragraph.</p>
<p id="p2">This is another paragraph.</p>
</div>
<script>
var para=document.createElement("p");
var node=document.createTextNode("This is new.");
para.appendChild(node);
var element=document.getElementById("div1");
var child=document.getElementById("p1");
element.insertBefore(para,child);
</script>
```

HTML DOM Elements (Nodes)

Removing Existing HTML Elements

To remove an HTML element, you must know the parent of the element.

```
<div id="div1">
<p id="p1">This is a paragraph.</p>
<p id="p2">This is another paragraph.</p>
</div>

<script>
var parent=document.getElementById("div1");
var child=document.getElementById("p1");
parent.removeChild(child);
</script>
```

HTML DOM Elements (Nodes)

Replacing HTML Elements

To replace an element to the HTML DOM, use the replaceChild() method.

```
<div id="div1">
<p id="p1">This is a paragraph.</p>
<p id="p2">This is another paragraph.</p>
</div>

<script>
var para=document.createElement("p");
var node=document.createTextNode("This is new.");
para.appendChild(node);

var parent=document.getElementById("div1");
var child=document.getElementById("p1");
parent.replaceChild(para,child);
</script>
```

HTML DOM Nodelist

A nodelist is an array of nodes (like an array of all HTML elements).

The getElementsByTagName() method returns a node list. A node list is an array of nodes.

The following code selects all <p> nodes in a document:

```
var x=document.getElementsByTagName("p");
```

The nodes can be accessed by index number. To access the second <p> you can write:

```
y=x[1];
```

HTML DOM Nodelist

HTML DOM Node List Length

The length property defines the number of nodes in a node-list.

You can loop through a node-list by using the length property:

```
x=document.getElementsByTagName("p");

for (i=0;i<x.length;i++)
{
document.write(x[i].innerHTML);
document.write("<br />");
}
```

More Info

More Info about JavaScript: http://www.w3schools.com/js/default.asp

Review

- Introduction of Http
  - Http
  - URLS
  - Https
- Introduction of Front-End Web Language
  - HTML
  - CSS
  - JavaScript

Review

- **Introduction of Http**
  - Http
  - URLS
  - Https
- **Introduction of Front-End Web Language**
  - HTML
  - CSS
  - JavaScript