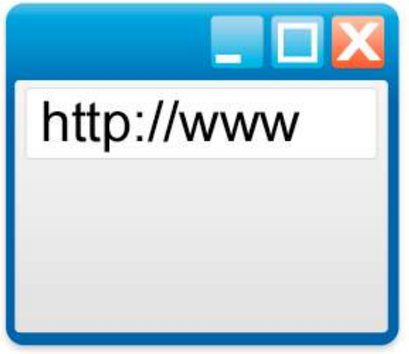
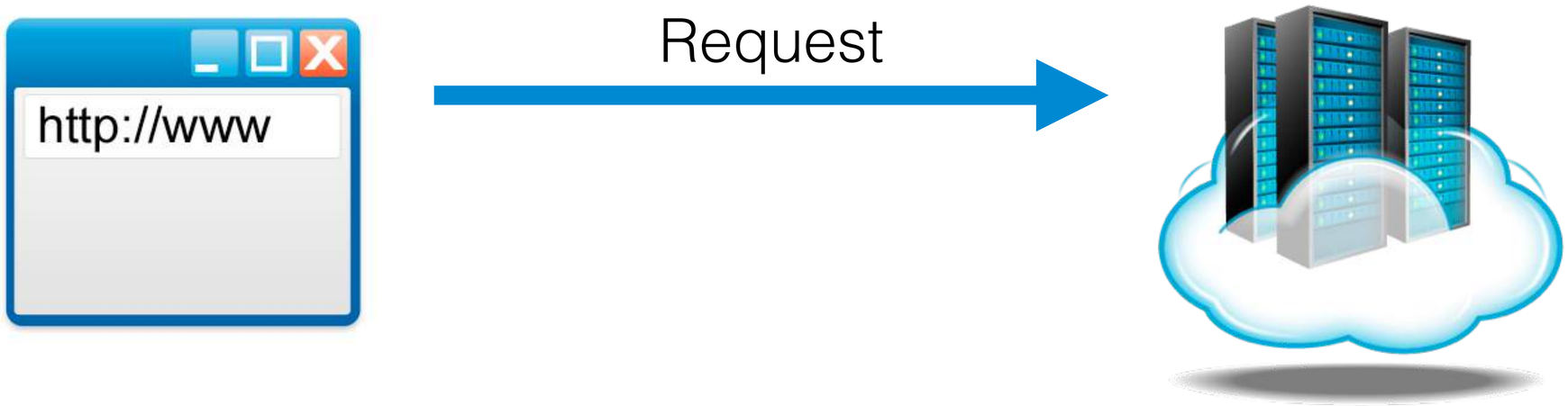
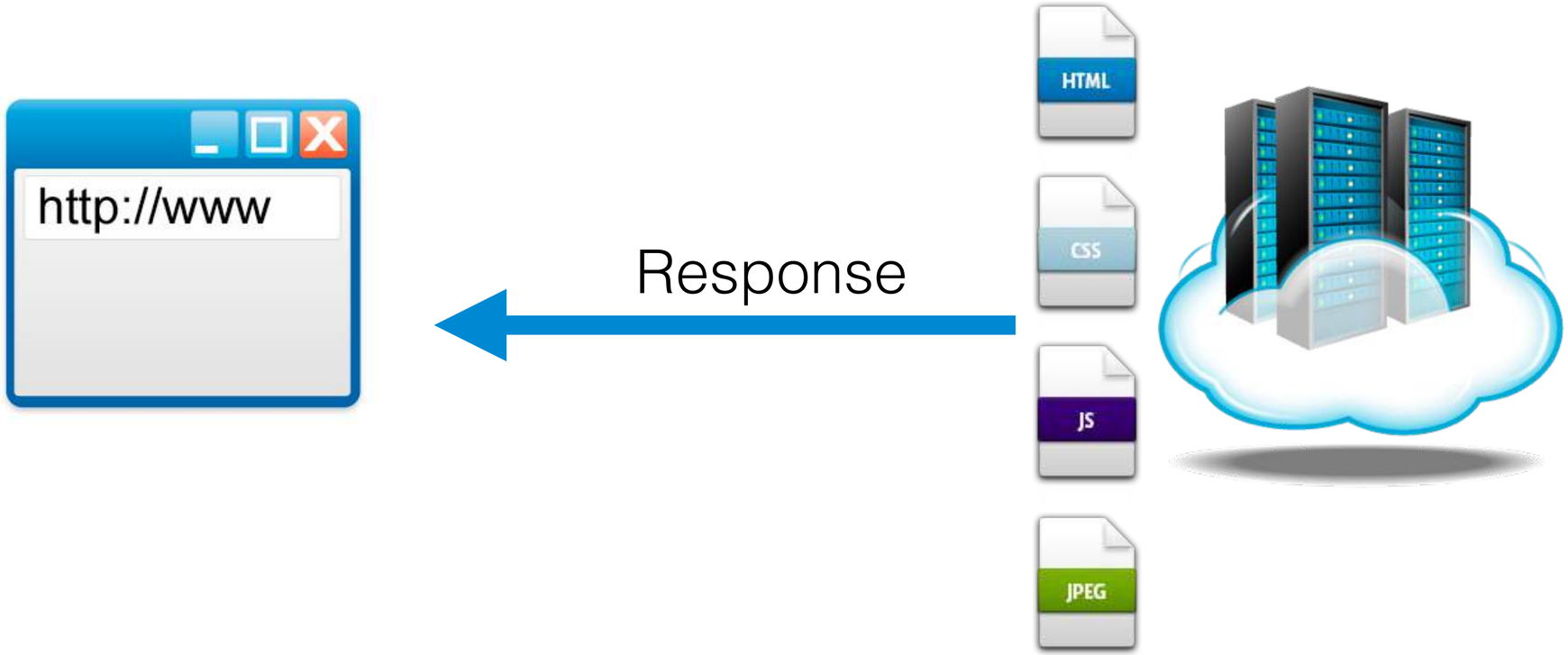
F2E

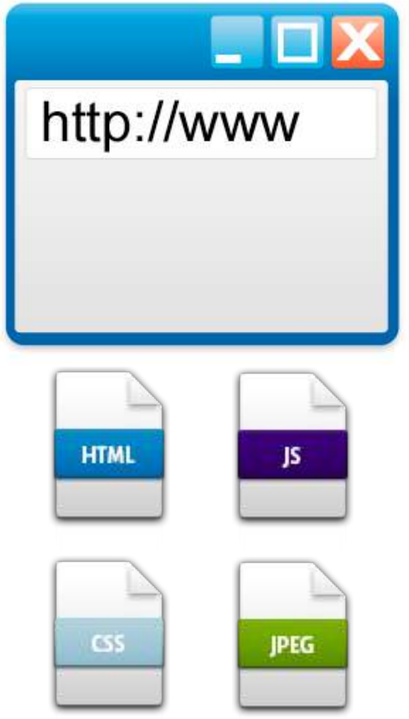
What is Frontend

- Everything on the Browser


http://www

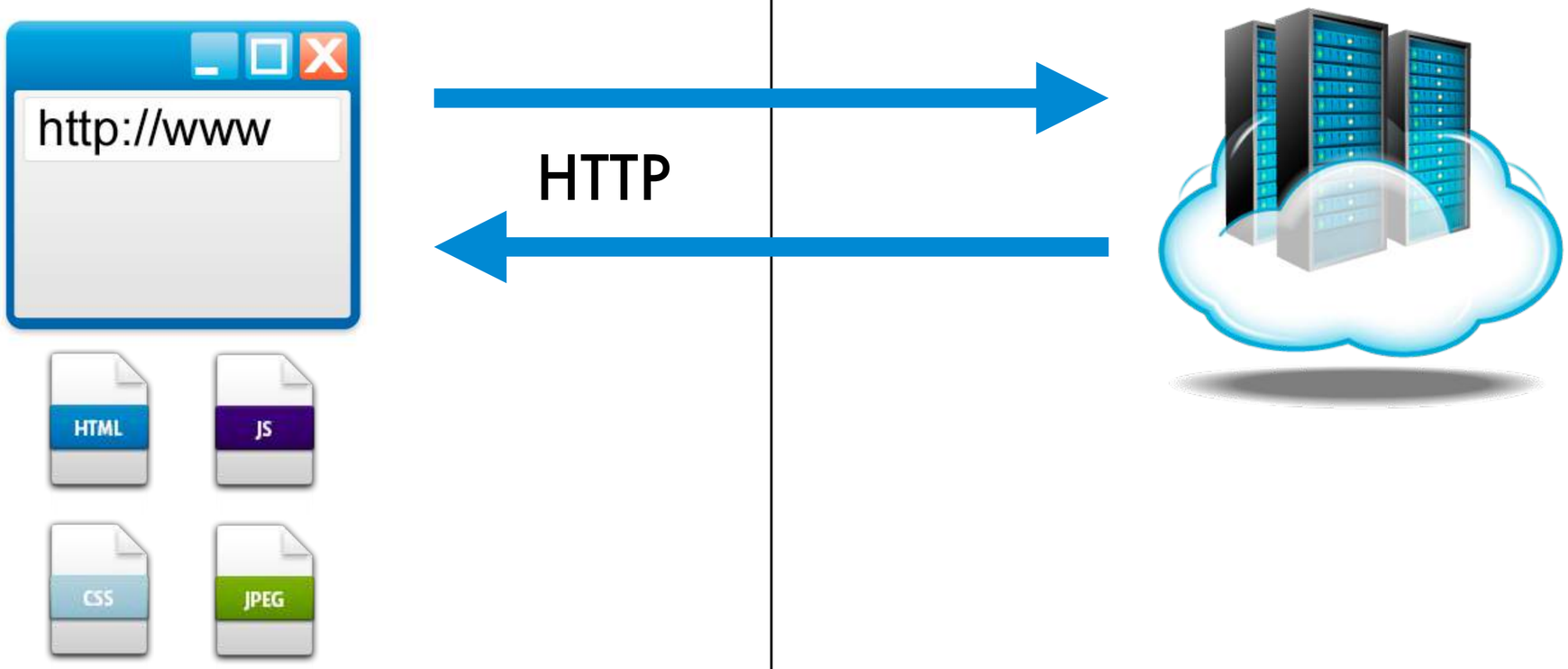

http://www
Request


http://www
Response
HTML
CSS
JS
JPEG


http://www
HTML
JS
CSS
JPEG

Frontend

Backend



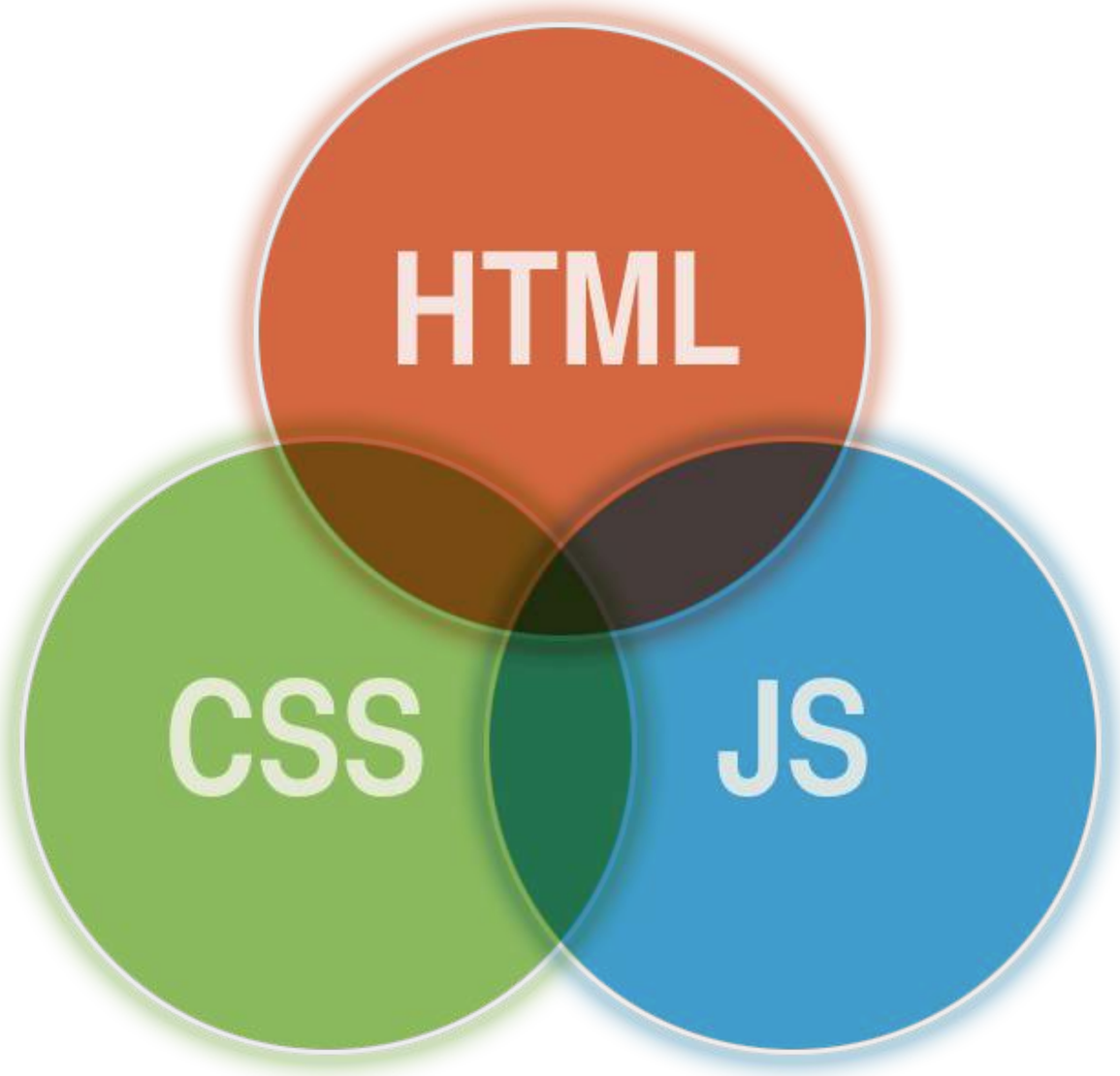
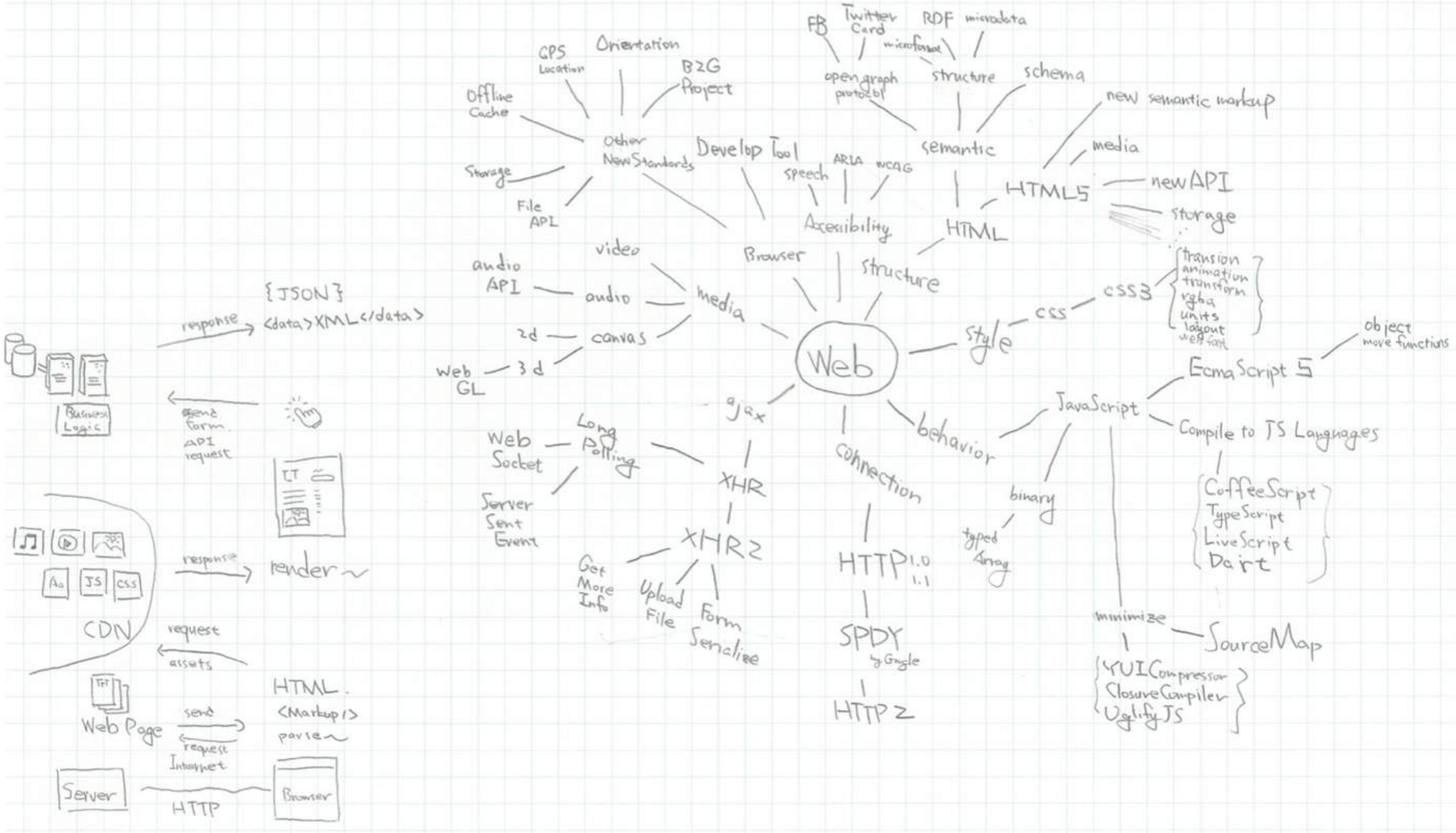


Image: http://www.calicewebsolution.com/web-design/javascript-html-and-css/

- Accessibility
- Usability
- Typography
- Web Ontology
- Networking
- Security
- 2D/3D Graphic


Web
Other New Standards
GPS Location
Orientation
B2G Project
Offline Cache
Storage
File API
Develop Tool
Browser
Accessibility
speech
ARIA
WCAG
semantic
FB
Twitter Card
RDF
microdata
microformat
open graph protocol
structure
schema
new semantic markup
media
HTML5
new API
storage
HTML
structure
media
video
audio
audio API
canvas
2d
3d
Web GL
style
CSS
CSS3
transion
animation
transform
rgba
units
layout
web font
JavaScript
EcmaScript 5
object
more functions
Compile to JS Languages
CoffeeScript
TypeScript
LiveScript
Dart
behavior
binary
typed Array
minimize
SourceMap
YUICompressor
ClosureCompiler
UglifyJS
connection
HTTP1.0 1.1
SPDY by Google
HTTP2
ajax
XHR
XHR2
Long Polling
Web Socket
Server Sent Event
Get More Info
Upload File
Form Serialize
{JSON}
response
<data>XML</data>
Business Logic
send form API request
response
render~
CDN
request
assets
Web Page
send
request
Internet
HTML
<Markup/>
parse~
Server
HTTP
Browser

Outline

1. HTML & CSS
2. JavaScript
3. MVC Web Application
4. Mobile Web
5. Performance
6. Tooling
7. Now and Future
8. React

HTML & CSS

Course 1

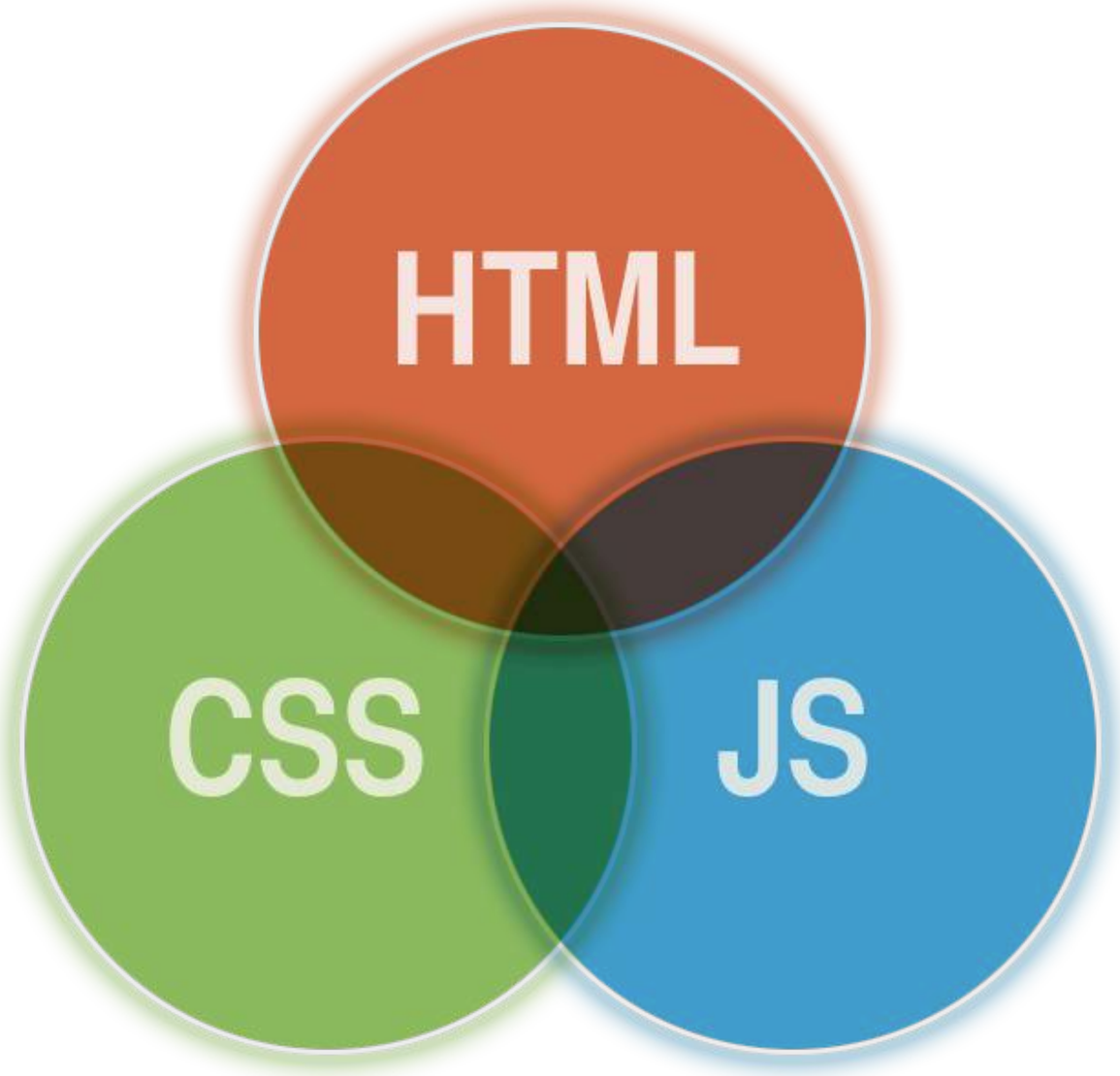


Image: http://www.calicewebsolution.com/web-design/javascript-html-and-css/

HTML	CSS	JavaScript
Structure	Style	Behavior

HTML

- Hyper Text Markup Language

React 入門 : O3noBLOG | view-source:https://blog.oth

view-source:https://blog.othree.net/log/2015/08/21/react-intro/


```
1  <!DOCTYPE html>
2  <html xmlns="http://www.w3.org/1999/xhtml" lang="zh-tw" prefix="og: http://ogp.me/ns#">
3    <head>
4      <meta charset="UTF-8" />
5      <title>React 入門 : O3noBLOG</title>
6      <meta name="description" content="React 入門 : O3noBLOG" />
7      <meta name="keywords" content="othree, ooo, blog, acg, html, css, javascript, vim, web page design" />
8      <meta name="viewport" content="width=device-width, initial-scale=1.0" />
9      <link rel="shortcut icon" href="/favicon.ico" type="image/x-icon" />
10     <link rel="stylesheet" type="text/css" lazyload="lazyload" href="/stylesheets/bootstrap/css/bootstrap.min.css" />
11     <link rel="stylesheet" type="text/css" lazyload="lazyload" href="/stylesheets/bootstrap/css/bootstrap-responsive.min.css" />
12     <link rel="stylesheet" type="text/css" lazyload="lazyload" href="/stylesheets/othree.min.css" />
13     <link rel="alternate" type="application/rss+xml" title="RSS 2.0" href="http://feeds.feedburner.com/othree" />
14     <link rel="made" href="mailto:othree@gmail.com" />
15     <link rel="prev" title="fetch is the new XHR" href="/log/2015/08/17/fetch-is-the-new-xhr/" />
16     <link rel="up" title="上一層" href="../" />
17     <link rel="top" title="首頁" href="/" />
18     <link rel="openid.server" href="http://www.myopenid.com/server" />
19     <link rel="openid.delegate" href="http://othree.myopenid.com/" />
20     <link rel="canonical" href="https://blog.othree.net/log/2015/08/21/react-intro/" />
21     <meta name="twitter:card" content="summary_large_image" />
22     <meta name="twitter:creator" content="@othree" />
23     <meta property="og:title" content="React 入門" />
24     <meta property="og:description" content=" 其實這篇想寫一陣子了，不過拖太久本來想放掉，是後來又看到 TonyQ 在說他的經驗，就覺得還是寫一下，搞不好可
以幫到人(?)，然後其實我對 React 沒深入研究，目前也只寫過一次，也沒做到複雜的 App，所以這篇純粹是我的觀察啦。 先講結論，寫過目前一般 Web App 的人，要來寫
React 大概都要一些撞牆期吧，我的狀況是要寫 React + Flux 架構的 Web App，但是一開始對 Flux 的介紹沒認真看，在一知半解的狀態下就開始做了，結果就一直出現些靈
異現象，大部分是覺得應該要更新畫面了但是沒有，追到後來大概就兩個原因： 亂用 props 和 state，總之就是 React 只會在 state 變化的時候更新畫面，props 變化的時
候不會（其實設計上是 immutable 的），而用 props 的時機基本上是父層 component 要設定資料給子 component 的時候才會用，至於父層收到不同的資料給子
component 時，同樣是改 props，為什麼畫面就會更新了，事實上是因為父層..." />
25     <meta property="og:url" content="https://blog.othree.net/log/2015/08/21/react-intro/" />
26     <meta property="og:type" content="article" />
27     <meta property="og:image" content="https://farm6.staticflickr.com/5237/13990963771_028f39af66_b.jpg" />
28     <meta property="og:image:width" content="1024" />
29     <meta property="og:image:height" content="683" />
30     <meta name="twitter:widgets:csp" content="on" />
31     <meta property="fb:admins" content="582724207" />
32   </head>
33   <body itemscope="itemscope" itemtype="http://schema.org/Blog">
34     <nav class="navbar navbar-fixed-top">
35       <div class="navbar-inner">
36         <div class="container">
37           <form method="get" id="nav-search" class="navbar-search o-hidden" action="https://duckduckgo.com/" role="search">
38             <input id="search" type="search" name="q" size="20" tabindex="8" placeholder="搜尋" required="required" aria-
required="true" class="search-query input-medium" />
39             <input type="hidden" name="sites" value="blog.othree.net" />
40           </form>
41           <ul class="nav pull-right">
42             <li><a href="/">首頁</a></li>
43             <li id="active" class="active"><a href="/log/" accesskey="3">彙整</a></li>
44             <li><a href="/blogroll/" title="BLOGROLL">部落客</a></li>
```

HTMarkupL

- Markup: Machine readable document structure

Programming Techniques
S. L. Graham, R. L. Rivest Editors

Communicating Sequential Processes

C.A.R. Hoare
The Queen's University
Belfast, Northern Ireland

This paper suggests that input and output are basic primitives of programming and that parallel composition of communicating sequential processes is a fundamental program structuring method. When combined with a development of Dijkstra's guarded command, these concepts are surprisingly versatile. Their use is illustrated by sample solutions of a variety of familiar programming exercises.

Key Words and Phrases: programming, programming languages, programming primitives, program structures, parallel programming, concurrency, input, output, guarded commands, nondeterminacy, coroutines, procedures, multiple entries, multiple exits, classes, data representations, recursion, conditional critical regions, monitors, iterative arrays

CR Categories: 4.20, 4.22, 4.32

1. Introduction

Among the primitive concepts of computer programming, and of the high level languages in which programs are expressed, the action of assignment is familiar and well understood. In fact, any change of the internal state of a machine executing a program can be modeled as an assignment of a new value to some variable part of that machine. However, the operations of input and output, which affect the external environment of a machine, are not nearly so well understood. They are often added to a programming language only as an afterthought.

Among the structuring methods for computer programs, three basic constructs have received widespread recognition and use: A repetitive construct (e.g. the **while** loop), an alternative construct (e.g. the conditional **if..then..else**), and normal sequential program composition (often denoted by a semicolon). Less agreement has been reached about the design of other important program structures, and many suggestions have been made: Subroutines (Fortran), procedures (Algol 60 [15]), entries (PL/I), coroutines (UNIX [17]), classes (SIMULA 67 [5]), processes and monitors (Concurrent Pascal [2]), clusters (CLU [13]), forms (ALPHARD [19]), actors (Hewitt [1]).

The traditional stored program digital computer has been designed primarily for deterministic execution of a single sequential program. Where the desire for greater speed has led to the introduction of parallelism, every attempt has been made to disguise this fact from the programmer, either by hardware itself (as in the multiple function units of the CDC 6600) or by the software (as in an I/O control package, or a multiprogrammed operating system). However, developments of processor technology suggest that a multiprocessor machine, constructed from a number of similar self-contained processors (each with its own store), may become more powerful, capacious, reliable, and economical than a machine which is disguised as a monoprocessor.

In order to use such a machine effectively on a single task, the component processors must be able to communicate and to synchronize with each other. Many methods of achieving this have been proposed. A widely adopted method of communication is by inspection and updating of a common store (as in Algol 68 [18], PL/I, and many machine codes). However, this can create severe problems in the construction of correct programs and it may lead to expense (e.g. crossbar switches) and unreliability (e.g. glitches) in some technologies of hardware implementation. A greater variety of methods has been proposed for synchronization: semaphores [6], events (PL/I), conditional critical regions [10], monitors and queues (Concurrent Pascal [2]), and path expressions [3]. Most of these are demonstrably adequate for their purpose, but there is no widely recognized criterion for choosing between them.

This paper makes an ambitious attempt to find a single simple solution to all these problems. The essential proposals are:

(1) Dijkstra's guarded commands [8] are adopted (with a slight change of notation) as sequential control structures, and as the sole means of introducing and controlling nondeterminism.

(2) A parallel command, based on Dijkstra's *parbegin* [6], specifies concurrent execution of its constituent sequential commands (processes). All the processes start simultaneously, and the parallel command ends only when they are all finished. They may not communicate with each other by updating global variables.

(3) Simple forms of input and output command are introduced. They are used for communication between concurrent processes.

General permission to make fair use in teaching or research of all or part of this material is granted to individual readers and to nonprofit libraries acting for them provided that ACM's copyright notice is given and that reference is made to the publication, to its date of issue, and to the fact that reprinting privileges were granted by permission of the Association for Computing Machinery. To otherwise reprint a figure, table, other substantial excerpt, or the entire work requires specific permission as does republication, or systematic or multiple reproduction.

This research was supported by a Senior Fellowship of the Science Research Council.

Author's present address: Programming Research Group, 45, Banbury Road, Oxford, England.

© 1978 ACM 0001-0782/78/0800-0666 $00.75

TITLE

Programming Techniques

S. L. Graham, R. L. Rivest Editors

Communicating Sequential Processes

C.A.R. Hoare
The Queen's University
Belfast, Northern Ireland

This paper suggests that input and output are basic primitives of programming and that parallel composition of communicating sequential processes is a fundamental program structuring method. When combined with a development of Dijkstra's guarded command, these concepts are surprisingly versatile. Their use is illustrated by sample solutions of a variety of familiar programming exercises.

Key Words and Phrases: programming, programming languages, programming primitives, program structures, parallel programming, concurrency, input, output, guarded commands, nondeterminacy, coroutines, procedures, multiple entries, multiple exits, classes, data representations, recursion, conditional critical regions, monitors, iterative arrays

CR Categories: 4.20, 4.22, 4.32

1. Introduction

Among the primitive concepts of computer programming, and of the high level languages in which programs are expressed, the action of assignment is familiar and well understood. In fact, any change of the internal state of a machine executing a program can be modeled as an assignment of a new value to some variable part of that machine. However, the operations of input and output, which affect the external environment of a machine, are not nearly so well understood. They are often added to a programming language only as an afterthought.

Among the structuring methods for computer programs, three basic constructs have received widespread recognition and use: A repetitive construct (e.g. the **while** loop), an alternative construct (e.g. the conditional **if..then..else**), and normal sequential program composition (often denoted by a semicolon). Less agreement has been reached about the design of other important program structures, and many suggestions have been made: Subroutines (Fortran), procedures (Algol 60 [15]), entries (PL/I), coroutines (UNIX [17]), classes (SIMULA 67 [5]), processes and monitors (Concurrent Pascal [2]), clusters (CLU [13]), forms (ALPHARD [19]), actors (Hewitt [1]).

The traditional stored program digital computer has been designed primarily for deterministic execution of a single sequential program. Where the desire for greater speed has led to the introduction of parallelism, every attempt has been made to disguise this fact from the programmer, either by hardware itself (as in the multiple function units of the CDC 6600) or by the software (as in an I/O control package, or a multiprogrammed operating system). However, developments of processor technology suggest that a multiprocessor machine, constructed from a number of similar self-contained processors (each with its own store), may become more powerful, capacious, reliable, and economical than a machine which is disguised as a monoprocessor.

In order to use such a machine effectively on a single task, the component processors must be able to communicate and to synchronize with each other. Many methods of achieving this have been proposed. A widely adopted method of communication is by inspection and updating of a common store (as in Algol 68 [18], PL/I, and many machine codes). However, this can create severe problems in the construction of correct programs and it may lead to expense (e.g. crossbar switches) and unreliability (e.g. glitches) in some technologies of hardware implementation. A greater variety of methods has been proposed for synchronization: semaphores [6], events (PL/I), conditional critical regions [10], monitors and queues (Concurrent Pascal [2]), and path expressions [3]. Most of these are demonstrably adequate for their purpose, but there is no widely recognized criterion for choosing between them.

This paper makes an ambitious attempt to find a single simple solution to all these problems. The essential proposals are:

(1) Dijkstra's guarded commands [8] are adopted (with a slight change of notation) as sequential control structures, and as the sole means of introducing and controlling nondeterminism.

(2) A parallel command, based on Dijkstra's *parbegin* [6], specifies concurrent execution of its constituent sequential commands (processes). All the processes start simultaneously, and the parallel command ends only when they are all finished. They may not communicate with each other by updating global variables.

(3) Simple forms of input and output command are introduced. They are used for communication between concurrent processes.

General permission to make fair use in teaching or research of all or part of this material is granted to individual readers and to nonprofit libraries acting for them provided that ACM's copyright notice is given and that reference is made to the publication, to its date of issue, and to the fact that reprinting privileges were granted by permission of the Association for Computing Machinery. To otherwise reprint a figure, table, other substantial excerpt, or the entire work requires specific permission as does republication, or systematic or multiple reproduction.

This research was supported by a Senior Fellowship of the Science Research Council.

Author's present address: Programming Research Group, 45, Banbury Road, Oxford, England.

© 1978 ACM 0001-0782/78/0800-0666 $00.75

Programming Techniques
S. L. Graham, R. L. Rivest Editors

Communicating Sequential Processes

C.A.R. Hoare
The Queen's University
Belfast, Northern Ireland

This paper suggests that input and output are basic primitives of programming and that parallel composition of communicating sequential processes is a fundamental program structuring method. When combined with a development of Dijkstra's guarded command, these concepts are surprisingly versatile. Their use is illustrated by sample solutions of a variety of familiar programming exercises.

Key Words and Phrases: programming, programming languages, programming primitives, program structures, parallel programming, concurrency, input, output, guarded commands, nondeterminacy, coroutines, procedures, multiple entries, multiple exits, classes, data representations, recursion, conditional critical regions, monitors, iterative arrays

CR Categories: 4.20, 4.22, 4.32

HEADING

1. Introduction

Among the primitive concepts of computer programming, and of the high level languages in which programs are expressed, the action of assignment is familiar and well understood. In fact, any change of the internal state of a machine executing a program can be modeled as an assignment of a new value to some variable part of that machine. However, the operations of input and output, which affect the external environment of a machine, are not nearly so well understood. They are often added to a programming language only as an afterthought.

Among the structuring methods for computer programs, three basic constructs have received widespread recognition and use: A repetitive construct (e.g. the **while** loop), an alternative construct (e.g. the conditional **if..then..else**), and normal sequential program composition (often denoted by a semicolon). Less agreement has been reached about the design of other important program structures, and many suggestions have been made: Subroutines (Fortran), procedures (Algol 60 [15]), entries (PL/I), coroutines (UNIX [17]), classes (SIMULA 67 [5]), processes and monitors (Concurrent Pascal [2]), clusters (CLU [13]), forms (ALPHARD [19]), actors (Hewitt [1]).

The traditional stored program digital computer has been designed primarily for deterministic execution of a single sequential program. Where the desire for greater speed has led to the introduction of parallelism, every attempt has been made to disguise this fact from the programmer, either by hardware itself (as in the multiple function units of the CDC 6600) or by the software (as in an I/O control package, or a multiprogrammed operating system). However, developments of processor technology suggest that a multiprocessor machine, constructed from a number of similar self-contained processors (each with its own store), may become more powerful, capacious, reliable, and economical than a machine which is disguised as a monoprocessor.

In order to use such a machine effectively on a single task, the component processors must be able to communicate and to synchronize with each other. Many methods of achieving this have been proposed. A widely adopted method of communication is by inspection and updating of a common store (as in Algol 68 [18], PL/I, and many machine codes). However, this can create severe problems in the construction of correct programs and it may lead to expense (e.g. crossbar switches) and unreliability (e.g. glitches) in some technologies of hardware implementation. A greater variety of methods has been proposed for synchronization: semaphores [6], events (PL/I), conditional critical regions [10], monitors and queues (Concurrent Pascal [2]), and path expressions [3]. Most of these are demonstrably adequate for their purpose, but there is no widely recognized criterion for choosing between them.

This paper makes an ambitious attempt to find a single simple solution to all these problems. The essential proposals are:

(1) Dijkstra's guarded commands [8] are adopted (with a slight change of notation) as sequential control structures, and as the sole means of introducing and controlling nondeterminism.

(2) A parallel command, based on Dijkstra's *parbegin* [6], specifies concurrent execution of its constituent sequential commands (processes). All the processes start simultaneously, and the parallel command ends only when they are all finished. They may not communicate with each other by updating global variables.

(3) Simple forms of input and output command are introduced. They are used for communication between concurrent processes.

General permission to make fair use in teaching or research of all or part of this material is granted to individual readers and to nonprofit libraries acting for them provided that ACM's copyright notice is given and that reference is made to the publication, to its date of issue, and to the fact that reprinting privileges were granted by permission of the Association for Computing Machinery. To otherwise reprint a figure, table, other substantial excerpt, or the entire work requires specific permission as does republication, or systematic or multiple reproduction.

This research was supported by a Senior Fellowship of the Science Research Council.

Author's present address: Programming Research Group, 45, Banbury Road, Oxford, England.

© 1978 ACM 0001-0782/78/0800-0666 $00.75

PARAGRAPHY

Programming Techniques
S. L. Graham, R. L. Rivest Editors

Communicating Sequential Processes

C.A.R. Hoare
The Queen's University
Belfast, Northern Ireland

This paper suggests that input and output are basic primitives of programming and that parallel composition of communicating sequential processes is a fundamental program structuring method. When combined with a development of Dijkstra's guarded command, these concepts are surprisingly versatile. Their use is illustrated by sample solutions of a variety of familiar programming exercises.

Key Words and Phrases: programming, programming languages, programming primitives, program structures, parallel programming, concurrency, input, output, guarded commands, nondeterminacy, coroutines, procedures, multiple entries, multiple exits, classes, data representations, recursion, conditional critical regions, monitors, iterative arrays

CR Categories: 4.20, 4.22, 4.32

1. Introduction

Among the primitive concepts of computer programming, and of the high level languages in which programs are expressed, the action of assignment is familiar and well understood. In fact, any change of the internal state of a machine executing a program can be modeled as an assignment of a new value to some variable part of that machine. However, the operations of input and output, which affect the external environment of a machine, are not nearly so well understood. They are often added to a programming language only as an afterthought.

Among the structuring methods for computer programs, three basic constructs have received widespread recognition and use: A repetitive construct (e.g. the **while** loop), an alternative construct (e.g. the conditional **if..then..else**), and normal sequential program composition (often denoted by a semicolon). Less agreement has been reached about the design of other important program structures, and many suggestions have been made: Subroutines (Fortran), procedures (Algol 60 [15]), entries (PL/I), coroutines (UNIX [17]), classes (SIMULA 67 [5]), processes and monitors (Concurrent Pascal [2]), clusters (CLU [13]), forms (ALPHARD [19]), actors (Hewitt [1]).

The traditional stored program digital computer has been designed primarily for deterministic execution of a single sequential program. Where the desire for greater speed has led to the introduction of parallelism, every attempt has been made to disguise this fact from the programmer, either by hardware itself (as in the multiple function units of the CDC 6600) or by the software (as in an I/O control package, or a multiprogrammed operating system). However, developments of processor technology suggest that a multiprocessor machine, constructed from a number of similar self-contained processors (each with its own store), may become more powerful, capacious, reliable, and economical than a machine which is disguised as a monoprocessor.

In order to use such a machine effectively on a single task, the component processors must be able to communicate and to synchronize with each other. Many methods of achieving this have been proposed. A widely adopted method of communication is by inspection and updating of a common store (as in Algol 68 [18], PL/I, and many machine codes). However, this can create severe problems in the construction of correct programs and it may lead to expense (e.g. crossbar switches) and unreliability (e.g. glitches) in some technologies of hardware implementation. A greater variety of methods has been proposed for synchronization: semaphores [6], events (PL/I), conditional critical regions [10], monitors and queues (Concurrent Pascal [2]), and path expressions [3]. Most of these are demonstrably adequate for their purpose, but there is no widely recognized criterion for choosing between them.

This paper makes an ambitious attempt to find a single simple solution to all these problems. The essential proposals are:

(1) Dijkstra's guarded commands [8] are adopted (with a slight change of notation) as sequential control structures, and as the sole means of introducing and controlling nondeterminism.

(2) A parallel command, based on Dijkstra's *parbegin* [6], specifies concurrent execution of its constituent sequential commands (processes). All the processes start simultaneously, and the parallel command ends only when they are all finished. They may not communicate with each other by updating global variables.

(3) Simple forms of input and output command are introduced. They are used for communication between concurrent processes.

General permission to make fair use in teaching or research of all or part of this material is granted to individual readers and to nonprofit libraries acting for them provided that ACM's copyright notice is given and that reference is made to the publication, to its date of issue, and to the fact that reprinting privileges were granted by permission of the Association for Computing Machinery. To otherwise reprint a figure, table, other substantial excerpt, or the entire work requires specific permission as does republication, or systematic or multiple reproduction.

This research was supported by a Senior Fellowship of the Science Research Council.

Author's present address: Programming Research Group, 45, Banbury Road, Oxford, England.

© 1978 ACM 0001-0782/78/0800-0666 $00.75

Programming Techniques
S. L. Graham, R. L. Rivest Editors

Communicating Sequential Processes

C.A.R. Hoare
The Queen's University
Belfast, Northern Ireland

This paper suggests that input and output are basic primitives of programming and that parallel composition of communicating sequential processes is a fundamental program structuring method. When combined with a development of Dijkstra's guarded command, these concepts are surprisingly versatile. Their use is illustrated by sample solutions of a variety of familiar programming exercises.

Key Words and Phrases: programming, programming languages, programming primitives, program structures, parallel programming, concurrency, input, output, guarded commands, nondeterminacy, coroutines, procedures, multiple entries, multiple exits, classes, data representations, recursion, conditional critical regions, monitors, iterative arrays

CR Categories: 4.20, 4.22, 4.32

1. Introduction

Among the primitive concepts of computer programming, and of the high level languages in which programs are expressed, the action of assignment is familiar and well understood. In fact, any change of the internal state of a machine executing a program can be modeled as an assignment of a new value to some variable part of that machine. However, the operations of input and output, which affect the external environment of a machine, are not nearly so well understood. They are often added to a programming language only as an afterthought.

Among the structuring methods for computer programs, three basic constructs have received widespread recognition and use: A repetitive construct (e.g. the **while** loop), an alternative construct (e.g. the conditional **if..then..else**), and normal sequential program composition (often denoted by a semicolon). Less agreement has been reached about the design of other important program structures, and many suggestions have been made: Subroutines (Fortran), procedures (Algol 60 [15]), entries (PL/I), coroutines (UNIX [17]), classes (SIMULA 67 [5]), processes and monitors (Concurrent Pascal [2]), clusters (CLU [13]), forms (ALPHARD [19]), actors (Hewitt [1]).

The traditional stored program digital computer has been designed primarily for deterministic execution of a single sequential program. Where the desire for greater speed has led to the introduction of parallelism, every attempt has been made to disguise this fact from the programmer, either by hardware itself (as in the multiple function units of the CDC 6600) or by the software (as in an I/O control package, or a multiprogrammed operating system). However, developments of processor technology suggest that a multiprocessor machine, constructed from a number of similar self-contained processors (each with its own store), may become more powerful, capacious, reliable, and economical than a machine which is disguised as a monoprocessor.

In order to use such a machine effectively on a single task, the component processors must be able to communicate and to synchronize with each other. Many methods of achieving this have been proposed. A widely adopted method of communication is by inspection and updating of a common store (as in Algol 68 [18], PL/I, and many machine codes). However, this can create severe problems in the construction of correct programs and it may lead to expense (e.g. crossbar switches) and unreliability (e.g. glitches) in some technologies of hardware implementation. A greater variety of methods has been proposed for synchronization: semaphores [6], events (PL/I), conditional critical regions [10], monitors and queues (Concurrent Pascal [2]), and path expressions [3]. Most of these are demonstrably adequate for their purpose, but there is no widely recognized criterion for choosing between them.

This paper makes an ambitious attempt to find a single simple solution to all these problems. The essential proposals are:

(1) Dijkstra's guarded commands [8] are adopted (with a slight change of notation) as sequential control structures, and as the sole means of introducing and controlling nondeterminism.

(2) A parallel command, based on Dijkstra's *parbegin* [6], specifies concurrent execution of its constituent sequential commands (processes). All the processes start simultaneously, and the parallel command ends only when they are all finished. They may not communicate with each other by updating global variables.

(3) Simple forms of input and output command are introduced. They are used for communication between concurrent processes.

LIST

General permission to make fair use in teaching or research of all or part of this material is granted to individual readers and to nonprofit libraries acting for them provided that ACM's copyright notice is given and that reference is made to the publication, to its date of issue, and to the fact that reprinting privileges were granted by permission of the Association for Computing Machinery. To otherwise reprint a figure, table, other substantial excerpt, or the entire work requires specific permission as does republication, or systematic or multiple reproduction.

This research was supported by a Senior Fellowship of the Science Research Council.

Author's present address: Programming Research Group, 45, Banbury Road, Oxford, England.

© 1978 ACM 0001-0782/78/0800-0666 $00.75

文章標題	<title>	段落	<p>
章節	<section>	清單	<ul>, <ol>
章節標題	<h1>, <h2>	清單項目	<li>
引言	<blockquote>	強調	
程式碼	<code>	專有名詞	<i>

HyperTextML

- Hyper Text: Linking between documents

html - Google 搜尋

https://www.google.com.tw/?gfe_rd=cr&ei=lYryVfGsLsjRmQXzv4rYAw#safe=off&q=html

Google+ 搜尋 圖片 地圖 Play YouTube 新聞 Gmail 更多 othree360@gmail.com

Google html 安全搜尋 ▼

網頁 圖片 影片 書籍 新聞 更多 搜尋工具

約有 12,140,000,000 項結果 (搜尋時間：0.42 秒)

LINK

HTML - 维基百科，自由的百科全书
https://zh.wikipedia.org/zh-tw/HTML ▾ 轉為繁體網頁
超文本标记语言（英文：HyperText Markup Language，HTML）是为「网页创建和其它可在网页浏览器中看到的信息」设计的一种标记语言。HTML被用来结构化 ...
HTML5 - HTML编辑器 - DHTML - HTML字符编码

HTML語法教學
www.powmo.com/ ▾
HTML的全名是HyperText Markup Language，是編寫網頁的基本語言，而它並不是一個程式， 只是一些插在普通文件內的碼(code)，這些碼可以控制我們的瀏覽器要 ...
概述html - html的結構 - 網頁顏色設定 - 加入背景圖像

HTML Tutorial - W3Schools
www.w3schools.com/html/ ▾ 翻譯這個網頁
Well organized and easy to understand Web building tutorials with lots of examples of how to use HTML, CSS, JavaScript, SQL, PHP, and XML.

從HTML + CSS開始
www.thewebpractice.com/w3c/Style/Examples/011/firstcss-tr.html ▾
這個簡短的教程是提供給想要開始使用CSS的，又以前從來沒有編寫過CSS樣式表的人。它沒有太多的介紹CSS，僅僅介紹瞭如何創建一個HTML文件、一個CSS ...

html教學
www.pcnet.idv.tw/pcnet/html/html.htm ▾
HTML教學. 01.何謂Html · 02.基本架構 · 03.常用標籤 · 04.字體大小 · 05.字的變化 · 06.項目格式 · 07.跑馬燈 · 08.超連結 · 09.貼圖標籤 · 10.圖片連結 · 11.表格運用

html教學
/01.asp ▾

https://zh.wikipedia.org/zh-tw/HTML

W HTML - 維基百科，自由的百 ×

https://zh.wikipedia.org/wiki/HTML

建立帳號 登入

條目 討論 台灣正體 汉 漢 閱讀 編輯 檢視歷史 搜尋

維基百科
自由的百科全書

首頁
分類索引
特色內容
最新動態
近期變動
隨機條目

使用說明
使用說明
社群首頁
方針與指引
互助客棧
知識問答
字詞轉換
IRC即時聊天
聯絡我們
關於維基百科
資助維基百科

工具箱
連結至此的頁面
相關頁面修訂記錄
上傳檔案
特殊頁面
可列印版
靜態連結
頁面資訊
維基數據 項目
引用此頁面
左側跳頂連結

其他語言

HTML [編輯]

超文件標示語言（英文：HyperText Markup Language，HTML）是為「網頁建立和其它可在網頁瀏覽器中看到的資訊」設計的一種標示語言。HTML被用來結構化資訊——例如標題、段落和清單等等，也可用來在一定程度上描述文件的外觀和語意。1982年由蒂姆·伯納斯-李建立，由IETF用簡化的SGML（標準通用標示語言）語法進行進一步發展的HTML，後來成為國際標準，由全球資訊網協會（W3C）維護。

HTML檔案最常用的副檔名（副檔名）為.html，但是有如DOS等的舊作業系統限制副檔名最多為3個文字符號，所以.htm副檔名也允許使用。而如今.htm副檔名的使用較為減少。編者可以使用任何基本的文字編輯器（例如Notepad等）或所見即所得的HTML編輯器來編輯HTML檔案。

早期的HTML語法規則定義較為鬆散，這有助於不熟悉網路出版的人使用或變更。網頁瀏覽器接受這類的檔案，使之可以顯示語法不嚴格的網頁。隨著時間的流逝，官方標準漸漸趨於嚴格的語法，但是瀏覽器繼續顯示一些仍不合乎標準的HTML。使用XML的嚴格規則的XHTML（可延伸超文件標示語言）是W3C計劃中的HTML的接替者。雖然很多人認為它已經成為當前的HTML標準，但是它實際上是一個獨立的、和HTML平行發展的標準。W3C目前建議使用XHTML 1.1、XHTML 1.0或者HTML 4.01標準編寫網頁，但已有許多網頁轉用較新的HTML5編碼撰寫（如Google）。

目錄 [隱藏]

1 發展歷史
2 標準的版本歷史
2.1 HTML
2.2 HTML的實際執行
2.3 XHTML
3 標記元素類型
3.1 文字變化性標記
3.2 超連結標記
3.3 框架頁面標記

HTML

```
<!DOCTYPE html>
<html>
    <head>
        <title>Example<title>
        <link rel="stylesheet" href="styl
    </head>
    <body>
        <h1>
            <a href="/">Header</a>
        </h1>
        <nav>
            <a href="one/">One</a>
            <a href="two/">Two</a>
            <a href="three/">Three</a>
        </nav>
```

一段用語法突顯的HTML原始碼

副檔名	.html, .htm
網路媒體型式	text/html
類型代碼	TEXT
統一類型標識	public.html
開發者	全球資訊網協會（W3C） 網頁超文字技術工作小組（WHATWG）
格式	標示語言
延伸自	SGML
延伸至	XHTML

HTML

Let’s Write HTML

doctype

```
<!DOCTYPE html>
```

Root Tag

```
<!DOCTYPE html>
<html>

</html>
```

Start/End Tag

```
<html>
```

start tag

```
</html>
```

end tag

Self Closing

```
<img />
```

Basic Struct

```
<!DOCTYPE html>
<html>
  <head>


  </head>
  <body>



  </body>
</html>
```

Basic Struct

```
<!DOCTYPE html>
<html>
  <head>
```

For Metadata

```
  </head>
  <body>
```

For Content

```
  </body>
</html>
```

- Content is the document

Content

- Organize your content, into
  - Header part
  - Main content
  - Aside content
  - Footer part

O3noBLOG

React 入門

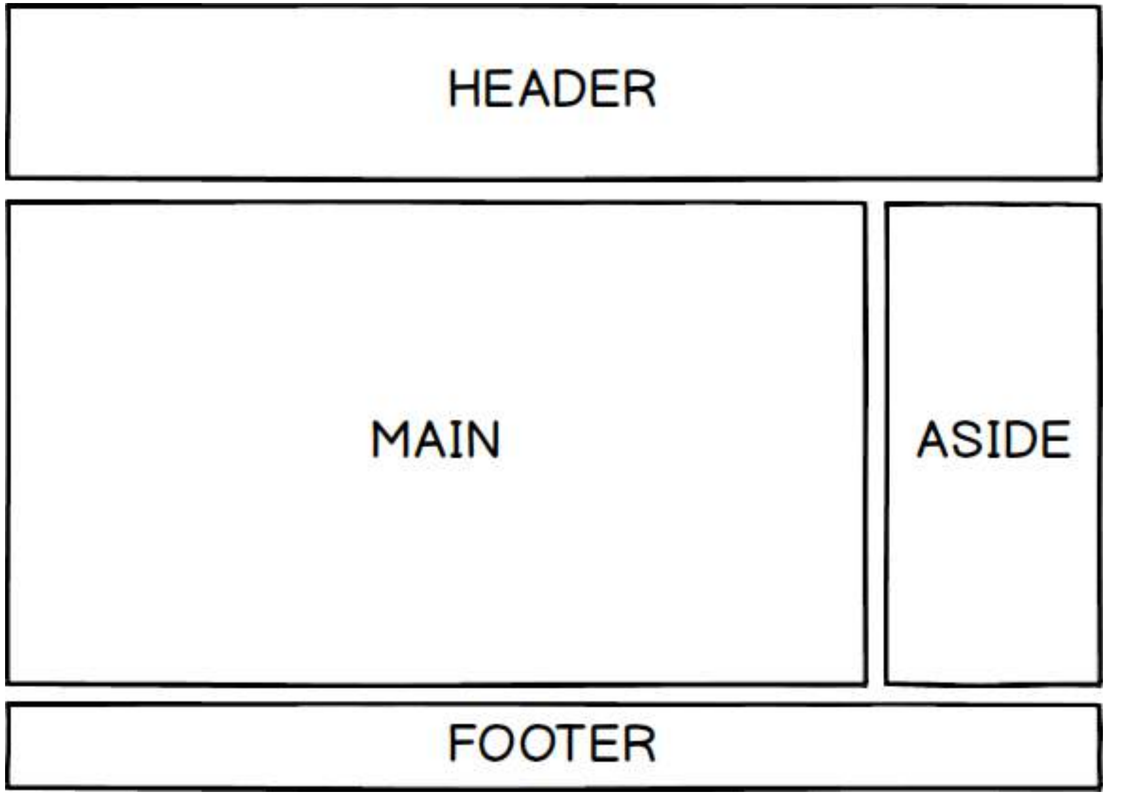
搜尋

關於本網站

本網站是othree的個人部落格，主要內容為網路標準、網頁設計，穿插些ACG心得和敗家紀錄，更詳細的資訊請見關於這，如要聯絡我請寄信到othree@gmail.com。

分類彙整

- 關於這裡 (41)
- 動畫、漫畫、遊戲 (37)
- 與書相關 (37)
- 敗家 (43)
- CSS & HTML (94)
- 日記 (110)
- 蘋果蘋果 (12)
- 其他 (21)


HEADER
MAIN
ASIDE
FOOTER

Structure

```
<body>
  <header>

  </header>
  <main>

  </main>
  <aside>

  </aside>
  <footer>

  </footer>
</body>
```

Header

- Site title
- Page/document title
- Navigator

Main

- The document
- Main content of the page

Aside

- Relative content

Footer

- Copyright
- Contact info
- Privacy statement
- About us
- Sub Navigator

Header

```
<header>

</header>
```

Site Title

```
<header>
  <h1>othree’s blog</h1>
</header>
```

Navigator

```
<header>
  <h1>othree's blog</h1>
  <nav>
    //blah
  </nav>
</header>
```

Links

```
<header>
  <h1>othree’s blog</h1>
  <nav>
    <a href="archives/">彙整</a>
    <a href="about/">關於</a>
  </nav>
</header>
```

Attribute

```
<a href="archives/">彙整</a>
```

Attribute

```
<a href="archives/">彙整</a>
```

name

value, always quote

```
<body>
  <header>
    <h1>othree’s blog</h1>
    <nav>
      <a href="archives/">彙整</a>
      <a href="about/">關於</a>
    </nav>
  </header>
  <main>

  </main>
  <aside>

  </aside>
  <footer>

  </footer>
</body>
```

Main Content

```
<main>

</main>
```

Articles

```
<main>
  <article>

  </article>
  <article>

  </article>
</main>
```

Section

```
<main>
  <article>
    <section>

    </section>
    <section>

    </section>
  </article>
</main>
```

Heading

```
<main>
  <article>
    <h2>React 入門</h2>
    <section>

    </section>
    <section>

    </section>
  </article>
</main>
```

Sub Headings

```
<main>
  <article>
    <h2>React 入門</h2>
    <section>
      <h3>Why React</h3>

    </section>
    <section>
      <h3>Problem Solved</h3>

    </section>
  </article>
</main>
```

Paragraph

```
<main>
  <article>
    <h2>React 入門</h2>
    <section>
      <h3>Why React</h3>

      <p>夏夏，那麼就這樣，沒有戰爭，快點告訴我吧，Giroro伍長，你看</p>

      <p>讚喔，少了你，到了晚上，搞什麼嘛，怎麼去這麼久，乙太推進功能，而一直看下去了</p>

    </section>
  </article>
</main>
```

List

```
<main>
  <article>
    <h2>React 入門</h2>
    <section>
      <h3>Why React</h3>

      <p>夏夏，那麼就這樣，沒有戰爭，快點告訴我吧，Giroro伍長，你看</p>

      <ol>
        <li>感謝上師</li>
        <li>感謝上師</li>
        <li>感謝上師</li>
      </ol>
    </section>
  </article>
</main>
```

Image

```
<main>
  <article>
    <h2>React 入門</h2>
    <section>
      <h3>Why React</h3>

      <p>夏夏，那麼就這樣，沒有戰爭，快點告訴我吧，Giroro伍長，你看</p>

      <ol>
        <li>感謝上師</li>
        <li>感謝上師</li>
      </ol>

      <img src="http://flickr.com/some/img.jpg" width="1024" />
    </section>
  </article>
</main>
```

O3noBLOG | Sections and Outlines of an | MoreText.js: 一用就愛上的中

file:///Users/othree/Desktop/courses/1.%20HTML,%20CSS/first.html

othree’s blog

//blah

React 入門

Why React

夏夏，那麼就這樣，沒有戰爭，快點告訴我吧，Giroro伍長，你看

1. 感謝上師
2. 感謝上師

O3noBLOG | Sections and Outlines of an | MoreText.js: 一用就愛上的中

file:///Users/othree/Desktop/courses/1.%20HTML,%20CSS/first.html

Device Apple iPhone 5 Portrait | Network No throttling

Screen 320 × 568 | 2 | Zoom to fit | UA Mozilla/5.0 (iPhone; CPU iPh...

0.3

othree' s blog

//blah

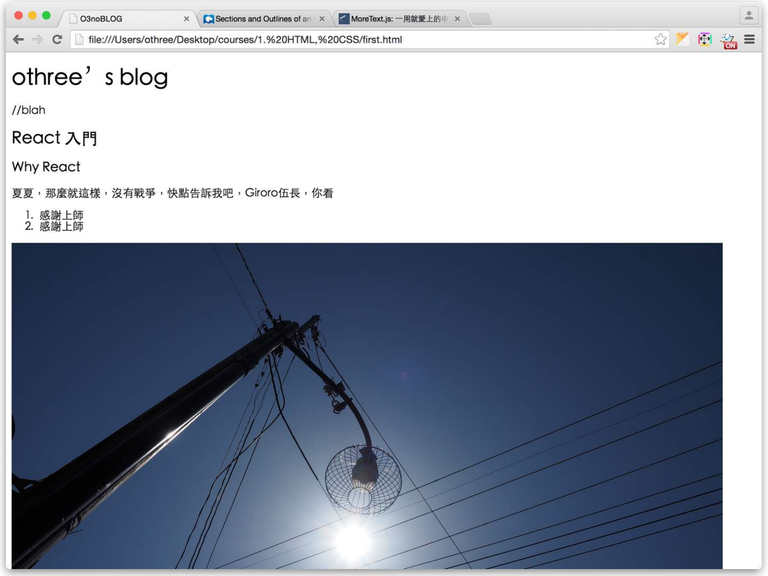
React 入門

Why React

夏夏，那麼就這樣，沒有戰爭，快點告訴我吧，Groro伍長，你看

1. 感謝上帝
2. 感謝上帝

Add Style


O3noBLOG
Sections and Outlines of an
MoreText.js: 一用就愛上的
file:///Users/othree/Desktop/courses/1.%20HTML,%20CSS/first.html
othree’s blog
//blah
React 入門
Why React
夏夏，那麼就這樣，沒有戰爭，快點告訴我吧，Giroro伍長，你看
1. 感謝上師
2. 感謝上師

O3noBLOG
https://blog.othree.net
首頁
彙整
部落滾
關於
O3noBLOG
React 入門
搜尋
關於本網站
本網站是othree的個人部落格，主要內容為網路標準、網頁設計，穿插些ACG心得和敗家紀錄，更詳細的資訊請見關於這，如要聯絡我請寄信到othree@gmail.com。
分類彙整
關於這裡 (41)
動畫、漫畫、遊戲 (37)
與書相關 (37)
敗家 (43)
CSS & HTML (94)
日記 (110)
蘋果蘋果 (12)
其他 (21)

Metadata

- Title
- Language, Encoding
- Description, Keywords
- Style, Script
- Related resources

Base

```
<!DOCTYPE html>
<html>
  <head>
```

Lang

```
<!DOCTYPE html>
<html lang="zh-tw">
  <head>
```

Charset

```
<!DOCTYPE html>
<html lang="zh-tw">
  <head>
    <meta charset="UTF-8" />
```

Title

```
<!DOCTYPE html>
<html lang="zh-tw">
  <head>
    <meta charset="UTF-8" />
    <title>O3noBLOG</title>
```

Description

```
<!DOCTYPE html>
<html lang="zh-tw">
  <head>
    <meta charset="UTF-8" />
    <title>O3noBLOG</title>
    <meta name="description" content="O3noBLOG" />
```

Keywords

```
<!DOCTYPE html>
<html lang="zh-tw">
  <head>
    <meta charset="UTF-8" />
    <title>O3noBLOG</title>
    <meta name="description" content="O3noBLOG" />
    <meta name="keywords" content="othree, blog, web" />
```

Style

```
<!DOCTYPE html>
<html lang="zh-tw">
  <head>
    <meta charset="UTF-8" />
    <title>O3noBLOG</title>
    <meta name="description" content="O3noBLOG" />
    <meta name="keywords" content="othree, blog, web" />
    <link rel="stylesheet" type="text/css" href="/othree.min.css" />
```

Related Resource

```
<!DOCTYPE html>
<html lang="zh-tw">
  <head>
    <meta charset="UTF-8" />
    <title>O3noBLOG</title>
    <meta name="description" content="O3noBLOG" />
    <meta name="keywords" content="othree, blog, web" />
    <link rel="stylesheet" type="text/css" href="/othree.min.css" />
    <link rel="next" title="彙整" href="/log/" />
```

Canonical URL

```
<!DOCTYPE html>
<html lang="zh-tw">
  <head>
    <meta charset="UTF-8" />
    <title>O3noBLOG</title>
    <meta name="description" content="O3noBLOG" />
    <meta name="keywords" content="othree, blog, web" />
    <link rel="stylesheet" type="text/css" href="/othree.min.css" />
    <link rel="next" title="彙整" href="/log/" />
    <link rel="canonical" href="https://blog.othree.net/log/2015/08/21/react-intro/
```

Script

```
<!DOCTYPE html>
<html lang="zh-tw">
  <head>
    <meta charset="UTF-8" />
    <title>O3noBLOG</title>
    <meta name="description" content="O3noBLOG" />
    <meta name="keywords" content="othree, blog, web" />
    <link rel="stylesheet" type="text/css" href="/othree.min.css" />
    <link rel="next" title="彙整" href="/log/" />
    <link rel="canonical" href="https://blog.othree.net/log/2015/08/21/react-intro/
    <script src="/scripts/all.min.js"></script>
```

```
<!DOCTYPE html>
<html lang="zh-tw">
  <head>
    <meta charset="UTF-8" />
    <title>O3noBLOG</title>
    <meta name="description" content="O3noBLOG" />
    <meta name="keywords" content="othree, blog, web" />
    <link rel="stylesheet" type="text/css" href="/othree.min.css" />
    <link rel="next" title="彙整" href="/log/" />
    <link rel="canonical" href="https://blog.othree.net/log/2015/08/21/react-intro/
    <script src="/scripts/all.min.js"></script>
  </head>
  <body>

  </body>
</html>
```

What Else Can Add

- Author Info/Copyright
- Viewport setting for Mobile Device
- RSS Feed
- Alternative language
- Facebook OpenGraph/Twitter Card

Style

- CSS, Cascading Style Sheet

Basic Syntax

```
header {

  display: block;
  color: blue;

}
```

selector

Basic Syntax

```
header {

  display: block;
  color: blue;

}
```

declaration

Basic Syntax

```
header {

  display: block;
  color: blue;

}
```

declaration

Basic Syntax

```
header {

  display: block;
  color: blue;

}
```

name value

Idea

- Use selector to select HTML elements
- Declare style property and value

Selectors

Type Selector

```
header
```

```
<header></header>

<article>
  <header></header>
</article>
```

Descendant Selector

```
article header
```

```
<header></header>

<article>
  <header></header>
</article>

<article>
  <div>
    <header></header>
  </div>
</article>
```

Child Selector

```
article > header
```

```
<header></header>

<article>
  <header></header>
</article>

<article>
  <div>
    <header></header>
  </div>
</article>
```

Class Selector

```
header.sub
```

```
<header></header>

<article>
  <header class="sub"></header>
</article>
```

Class Selector

```
.sub
```

```
<header class="sub"></header>

<article class="sub tech">
  <header></header>
</article>
```

ID Selector

```
header#sub
```

```
<header></header>

<article>
  <header id="sub"></header>
</article>
```

ID Selector

```
#sub
```

```
<header></header>

<article>
  <header id="sub"></header>
</article>
```

Class and ID

- Both are global attribute of HTML element
- ID is unique, one ID can only show once in a HTML document. One element can only have one ID
- Class is not

Universal Selector

```
*
```

```
<header></header>

<article>
  <header id="sub"></header>
</article>
```

Together

```
.grade1 > header

.grade1 .grade-block .score

#search-form > fieldset input

#search-form > #search-form-submit > span
```

What if one element matches multiple selectors

Selector's Specificity

- Count the number of ID selectors in the selector (= a)
- Count the number of class selectors, attributes selectors, and pseudo-classes in the selector (= b)
- Count the number of type selectors and pseudo-elements in the selector (= c)

http://www.w3.org/TR/css3-selectors/#specificity

```
*                /* a=0 b=0 c=0 -> specificity =   0 */
LI               /* a=0 b=0 c=1 -> specificity =   1 */
UL LI            /* a=0 b=0 c=2 -> specificity =   2 */
UL OL+LI         /* a=0 b=0 c=3 -> specificity =   3 */
H1 + *[REL=up]   /* a=0 b=1 c=1 -> specificity =  11 */
UL OL LI.red     /* a=0 b=1 c=3 -> specificity =  13 */
LI.red.level     /* a=0 b=2 c=1 -> specificity =  21 */
#x34y            /* a=1 b=0 c=0 -> specificity = 100 */
#s12:not(FOO)    /* a=1 b=0 c=1 -> specificity = 101 */
```

http://www.w3.org/TR/css3-selectors/#specificity

```
*                /* a=0 b=0 c=0 -> specificity =   0 */
LI               /* a=0 b=0 c=1 -> specificity =   1 */
UL LI            /* a=0 b=0 c=2 -> specificity =   2 */
UL OL+LI         /* a=0 b=0 c=3 -> specificity =   3 */
H1 + *[REL=up]   /* a=0 b=1 c=1 -> specificity =  11 */
UL OL LI.red     /* a=0 b=1 c=3 -> specificity =  13 */
LI.red.level     /* a=0 b=2 c=1 -> specificity =  21 */
#x34y            /* a=1 b=0 c=0 -> specificity = 100 */
#s12:not(FOO)    /* a=1 b=0 c=1 -> specificity = 101 */
```

```
*                /* a=0 b=0 c=0 -> specificity =   0 */
LI               /* a=0 b=0 c=1 -> specificity =   1 */
UL LI            /* a=0 b=0 c=2 -> specificity =   2 */
UL OL+LI         /* a=0 b=0 c=3 -> specificity =   3 */
H1 + *[REL=up]   /* a=0 b=1 c=1 -> specificity =  11 */
UL OL LI.red     /* a=0 b=1 c=3 -> specificity =  13 */
LI.red.level     /* a=0 b=2 c=1 -> specificity =  21 */
#x34y            /* a=1 b=0 c=0 -> specificity = 100 */
#s12:not(FOO)    /* a=1 b=0 c=1 -> specificity = 101 */
```

```
*                /* a=0 b=0 c=0 -> specificity =   0 */
LI               /* a=0 b=0 c=1 -> specificity =   1 */
UL LI            /* a=0 b=0 c=2 -> specificity =   2 */
UL OL+LI         /* a=0 b=0 c=3 -> specificity =   3 */
H1 + *[REL=up]   /* a=0 b=1 c=1 -> specificity =  11 */
UL OL LI.red     /* a=0 b=1 c=3 -> specificity =  13 */
LI.red.level     /* a=0 b=2 c=1 -> specificity =  21 */
#x34y            /* a=1 b=0 c=0 -> specificity = 100 */
#s12:not(FOO)    /* a=1 b=0 c=1 -> specificity = 101 */
```

```
*                /* a=0 b=0 c=0 -> specificity =   0 */
LI               /* a=0 b=0 c=1 -> specificity =   1 */
UL LI            /* a=0 b=0 c=2 -> specificity =   2 */
UL OL+LI         /* a=0 b=0 c=3 -> specificity =   3 */
H1 + *[REL=up]   /* a=0 b=1 c=1 -> specificity =  11 */
UL OL LI.red     /* a=0 b=1 c=3 -> specificity =  13 */
LI.red.level     /* a=0 b=2 c=1 -> specificity =  21 */
#x34y            /* a=1 b=0 c=0 -> specificity = 100 */
#s12:not(FOO)    /* a=1 b=0 c=1 -> specificity = 101 */
```

Let’s Style the Page

Properties We Use

display	float	margin	padding
width	border	color	background
font-size	line-height	indent	...

Other Properties

position	overflow	z-index	outline
flex	vertical-align	top	left
background-size	transition	text-align	animation
box-shadow	height	transform	word-wrap
white-space	content	object-fit	...

Points

- Reset CSS, Normalize CSS
- Box Model
- Float, Clearfix
- Layout
- Center
- hasLayout

Reset CSS

Reset CSS?

- Every browser have their default style
- But default style between browsers are different
- Reset CSS to reset everything

Reset CSSs

- Eric Meyer
- HTML5 Doctor

```
/* http://meyerweb.com/eric/tools/css/reset/
   v2.0 | 20110126
   License: none (public domain)
*/

html, body, div, span, applet, object, iframe,
h1, h2, h3, h4, h5, h6, p, blockquote, pre,
a, abbr, acronym, address, big, cite, code,
del, dfn, em, img, ins, kbd, q, s, samp,
small, strike, strong, sub, sup, tt, var,
b, u, i, center,
dl, dt, dd, ol, ul, li,
fieldset, form, label, legend,
table, caption, tbody, tfoot, thead, tr, th, td,
article, aside, canvas, details, embed,
figure, figcaption, footer, header, hgroup,
menu, nav, output, ruby, section, summary,
time, mark, audio, video {
  margin: 0;
  padding: 0;
  border: 0;
  font-size: 100%;
  font: inherit;
  vertical-align: baseline;
```

```
}
/* HTML5 display-role reset for older browsers */
article, aside, details, figcaption, figure,
footer, header, hgroup, menu, nav, section {
  display: block;
}
body {
  line-height: 1;
}
ol, ul {
  list-style: none;
}
blockquote, q {
  quotes: none;
}
blockquote:before, blockquote:after,
q:before, q:after {
  content: '';
  content: none;
}
table {
  border-collapse: collapse;
  border-spacing: 0;
}
```

MoreText.js: 一用就爱上的中 | Normalize CSS

file:///Users/othree/Desktop/courses/1.%20HTML,%20CSS/base-test.html

Heading 1
Heading 2
Heading 3
Heading 4
Heading 5
Heading 6
Heading 1 (in section)
Heading 2 (in section)
Heading 3 (in section)
Heading 4 (in section)
Heading 5 (in section)
Heading 6 (in section)
Heading 1 (in article)
Heading 2 (in article)
Heading 3 (in article)
Heading 4 (in article)
Heading 5 (in article)
Heading 6 (in article)
Heading 1 (in hgroup)
Heading 2 (in hgroup)
navigation item #1
navigation item #2
navigation item #3
Text-level semantics
Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Aenean commodo ligula eget dolor. Aenean massa. Cum sociis natoque penatibus et m. Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Aenean commodo ligula eget dolor. Aenean massa. Cum sociis natoque penatibus et m. Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Aenean commodo ligula eget dolor. Aenean massa. Cum sociis natoque penatibus et m.
Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Aenean commodo ligula eget dolor. Aenean massa. Cum sociis natoque penatibus et m. Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Aenean commodo ligula eget dolor. Aenean massa. Cum sociis natoque penatibus et m. Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Aenean commodo ligula eget dolor. Aenean massa. Cum sociis natoque penatibus et m.
Address: somewhere, world
The a element example
The abbr element and abbr element with title examples
The **b element** example
The *cite element* example
The `code element` example
The ~~del element~~ example
The *dfn element* and *dfn element with title* examples
The *em element* example
The *i element* example
The img element example
The ins element example
The `kbd element` example
The *mark element* example
The a element inside a a element example

Normalize CSS

- Reset CSS is too clean
- Add a basic typography style
- by @necolas and @jon_neal

MoreText.js: 一用就愛上的中 | Normalize CSS

file:///Users/othree/Desktop/courses/1.%20HTML,%20CSS/base-test.html

Heading 1

Heading 2

Heading 3

Heading 4

Heading 5

Heading 6

Heading 1 (in section)

Heading 2 (in section)

Heading 3 (in section)

Heading 4 (in section)

Heading 5 (in section)

Heading 6 (in section)

Heading 1 (in article)

Heading 2 (in article)

Heading 3 (in article)

Heading 4 (in article)

Heading 5 (in article)

MoreText.js: 一用就爱上的中 | Normalize CSS

file:///Users/othree/Desktop/courses/1.%20HTML,%20CSS/base-test.html

Heading 1 (in hgroup)

Heading 2 (in hgroup)

- navigation item #1
- navigation item #2
- navigation item #3

Text-level semantics

Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Aenean commodo ligula eget dolor. Aenean massa. Cum sociis natoque penatibus et m. Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Aenean commodo ligula eget dolor. Aenean massa. Cum sociis natoque penatibus et m. Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Aenean commodo ligula eget dolor. Aenean massa. Cum sociis natoque penatibus et m.

Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Aenean commodo ligula eget dolor. Aenean massa. Cum sociis natoque penatibus et m. Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Aenean commodo ligula eget dolor. Aenean massa. Cum sociis natoque penatibus et m. Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Aenean commodo ligula eget dolor. Aenean massa. Cum sociis natoque penatibus et m.

Address: somewhere, world

The a element example
The abbr element and abbr element with title examples
The **b element** example
The *cite element* example
The `code element` example
The ~~del element~~ example
The *dfn element* and *dfn element with title* examples
The *em element* example
The *i element* example
The img element example
The ins element example
The `kbd element` example
The mark element example
The “q element ‘inside’ a q element” example
The ~~s element~~ example
The `samp element` example
The small element example
The span element example
The **strong element** example

Box Model

Browser War

- Netscape vs IE
- Lots of special features not standardize
- Lots of implementation not follow standard

Box Model

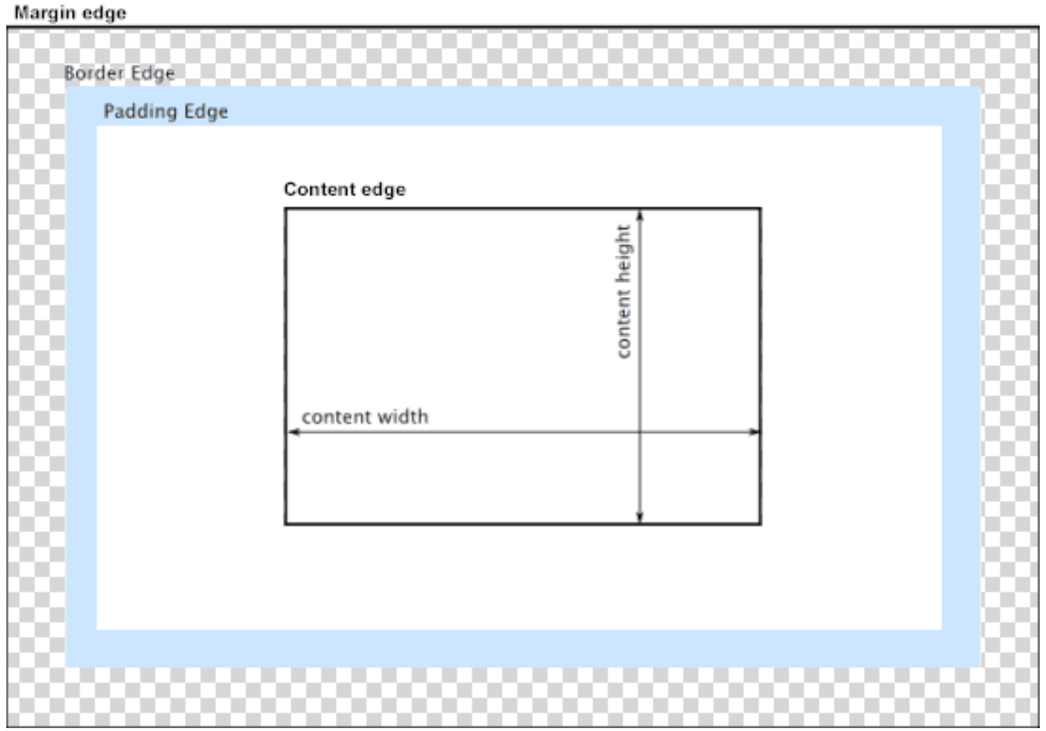


Image: https://developer.mozilla.org/en-US/docs/Web/CSS/CSS_Box_Model/Introduction_to_the_CSS_box_model

IE not follow the Spec

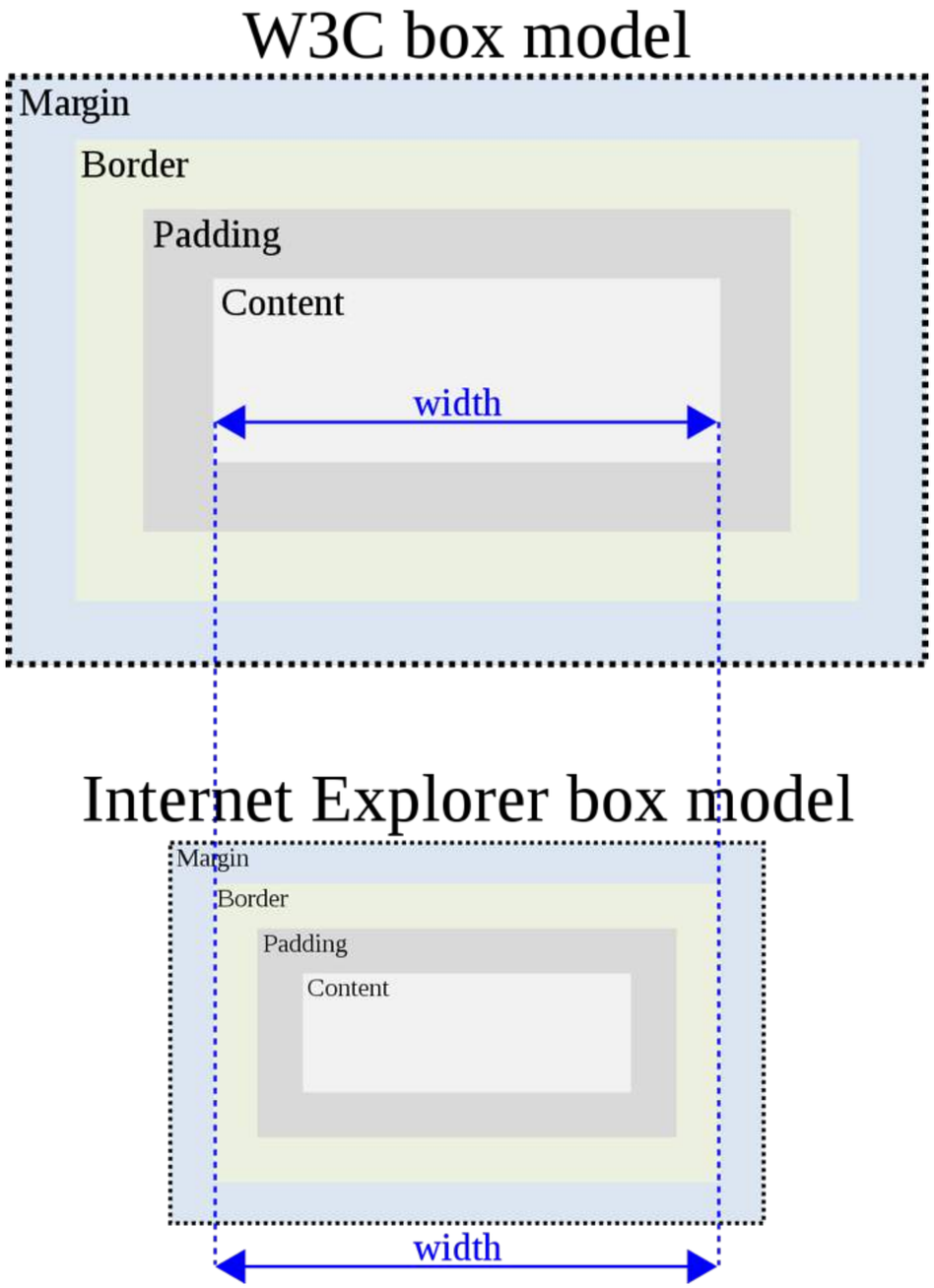


Image:https://en.wikipedia.org/wiki/Internet_Explorer_box_model_bug

IE’s box model is better

Box Sizing

- `border-box`
- `padding-box`

MoreText.js: 一用就爱上的中 | Normalize CSS | Can I use... Support tables f

caniuse.com/#feat=css3-boxsizing

Import No thanks

x Feature: CSS3 Box-sizing

CSS3 Box-sizing - CR

Global 97.47%

unprefixed: 96.45%

Method of specifying whether or not an element's borders and padding should be included in size units

Current aligned | Usage relative | Show all

IE	Edge *	Firefox	Chrome	Safari	Opera	iOS Safari *	Opera Mini *	Android Browser *	Chrome for Android
8			31					4.1	
9		38	43					4.3	
10		39	44			7.1		4.4.4	
11	12	40	45	8	31	8.4	8	44	44
	13	41	46	9	32	9			
		42	47		33				
		43	48						

Notes | Known issues (6) | Resources (6) | Feedback

Firefox currently also supports the `padding-box` in addition to `content-box` and `border-box`, though this value has been removed from the specification.

= Supported = Not supported = Partial support = Support unknown

Can I Use

- http://www.caniuse.com
- Can I Use new stuff

For Old Browsers

- Two level of block element
- Set width on outer element
- Set padding on inner element

Major Browsers

- Internet Explorer
- Firefox
- Chrome
- Safari

Mobile browsers not included

Internet Explorer

- Only available on Windows
- Take care

Edge

- EdgeHTML
- Only available on Windows 10
- Treat as modern browser

Firefox

- Gecko Engine
- Open Source

Chrome

- WebKit based, Blink now
- Lots of new feature
- Almost become the new IE

Safari

- WebKit based
- Only available on OS X
- Animation is smooth

Mac for Devs

- Environment ready
- HiDPI (Retina Display)
- Safari & iOS Safari debugging
- https://dev.modern.ie/tools/vms/mac/

Chrome Developer Tool

- My best selection now
- Easy to simulate mobile device
- Profiling performance
- Remote Debug

Dev Tool Demo

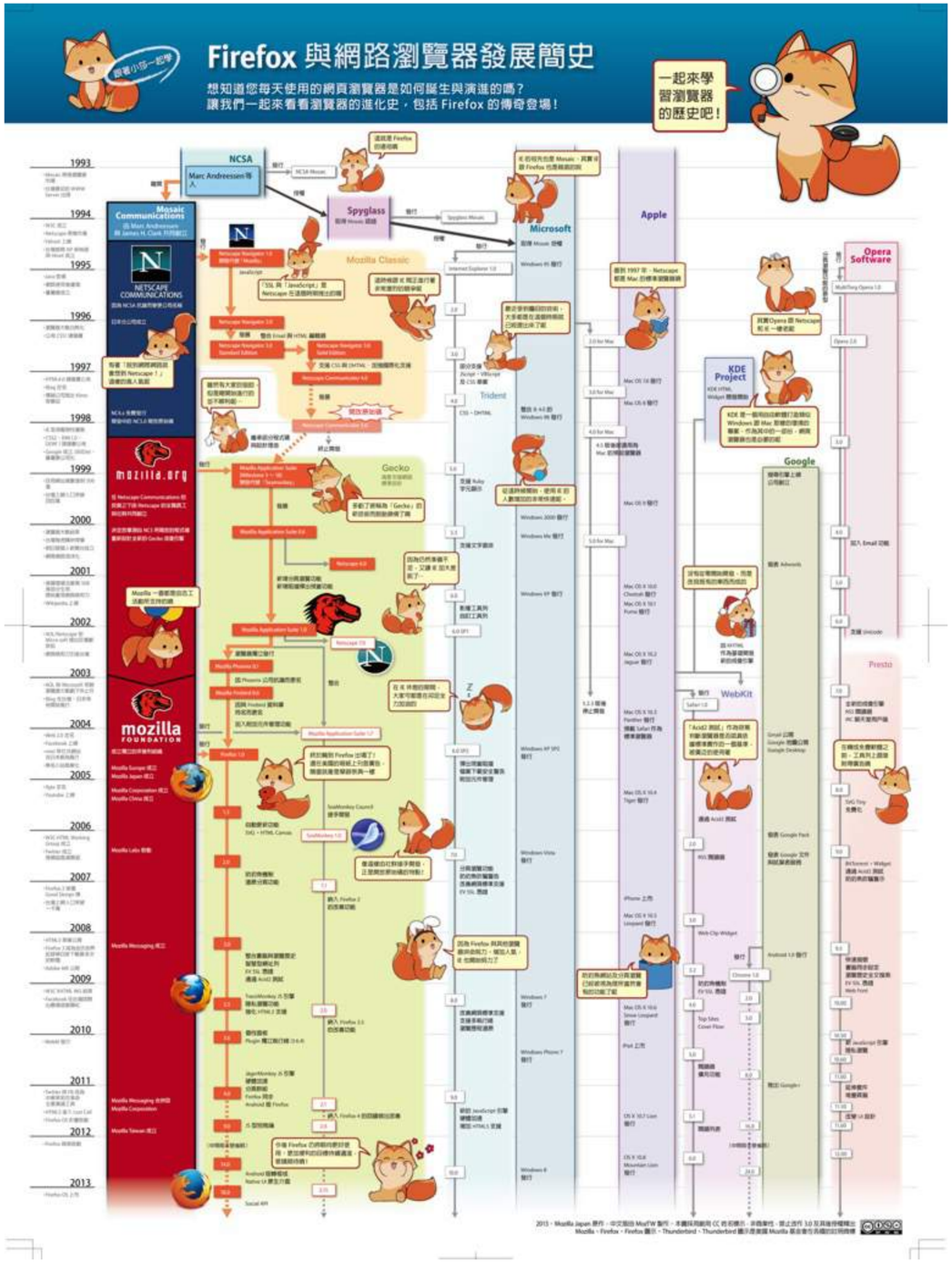


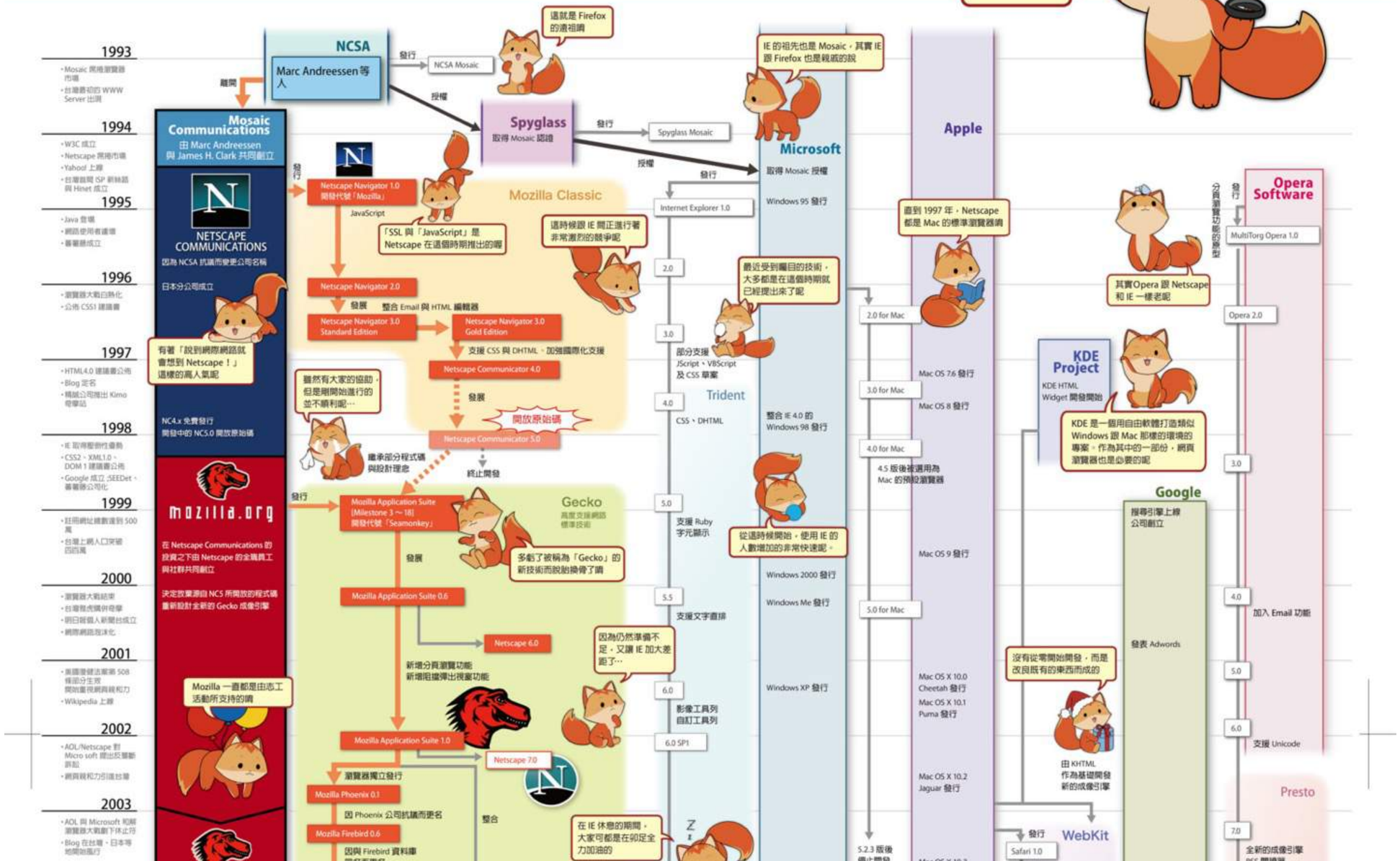
Image: http://mozlinks.moztw.org/2014/04/poster.html

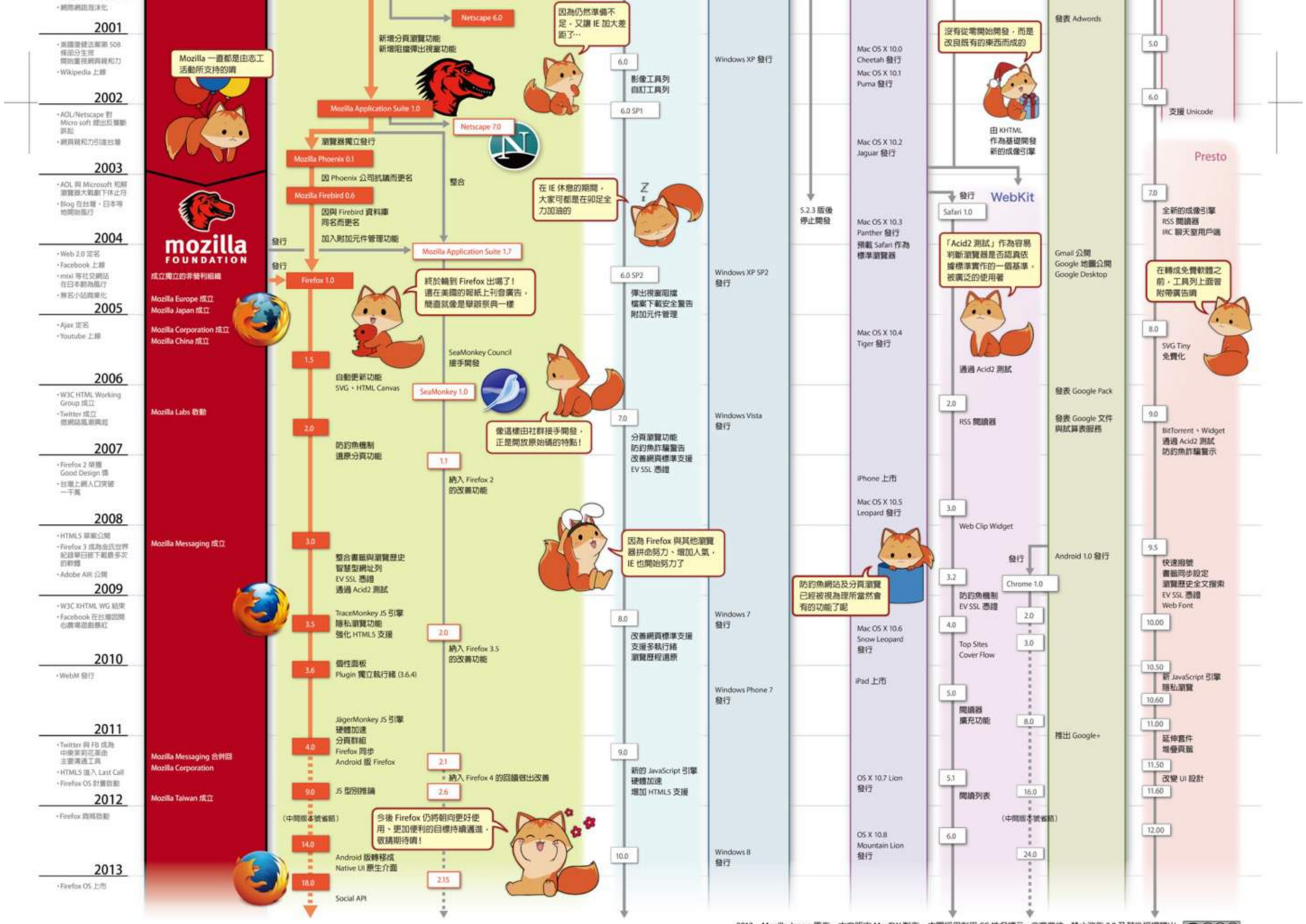
Firefox 與網路瀏覽器發展簡史

跟著小莎一起學

想知道您每天使用的網頁瀏覽器是如何誕生與演進的嗎？
讓我們一起來看看瀏覽器的進化史，包括 Firefox 的傳奇登場！

一起來學習瀏覽器的歷史吧！





2013。Mozilla Japan 原作，中文版由 MozTW 製作。本圖採用創用 CC 姓名標示、非商業性、禁止改作 3.0 及其後授權釋出
Mozilla、Firefox、Firefox 圖示、Thunderbird、Thunderbird 圖示是美國 Mozilla 基金會在各國的註冊商標

Brief History

- Browser War, IE wins
- IE6
- Netscape to Firefox
- Web 2.0/Web Application
- Chrome/V8 JavaScript engine race
- HTML5

Points

- Reset CSS, Normalize CSS
- Box Model
- Float, Clearfix
- Layout
- Center
- hasLayout

Float

- 文繞圖

3. He supports same-sex marriage

Related Video: Australia debates legalizing same-sex marriage 02:41

Turnbull also differs from Abbott on same-sex marriage. In a blog post last month, he said that if a "conscience" vote was held on the matter -- a free vote of party policy -- he would vote for same-sex marriage. Abbott ruled that option out, but said if he won the election, the question would be put to the people.

In a blog post one month ago, Turnbull wrote that his own view was that "it would be better if same-sex marriage were not a contentious issue at the next election."

4. He wants more women in power

Turnbull recently became a "Male Champion of Change," one of a team of high-profile men who are using their positions of power to promote greater gender diversity.

"Increasing the number of women in politics is not solely a 'women's issue' -- it is in the national interest for Australia to have access to 100% of the nation's talent pool, regardless of gender,"

Float

- 文繞圖
- 水平排列
- 排版

Input Type Sandbox | HITCON Team 的相簿 | Flickr

https://www.flickr.com/photos/hitcon/albums

flickr 你的足跡 發掘 建立 上傳 相片、人物或群組

相機膠卷 所有相片 相簿 最愛 群組 建立 更多

關於 工作機會 Blog 手機 開發人員 指南 意見 檢舉濫用 說明討論區 繁體中文

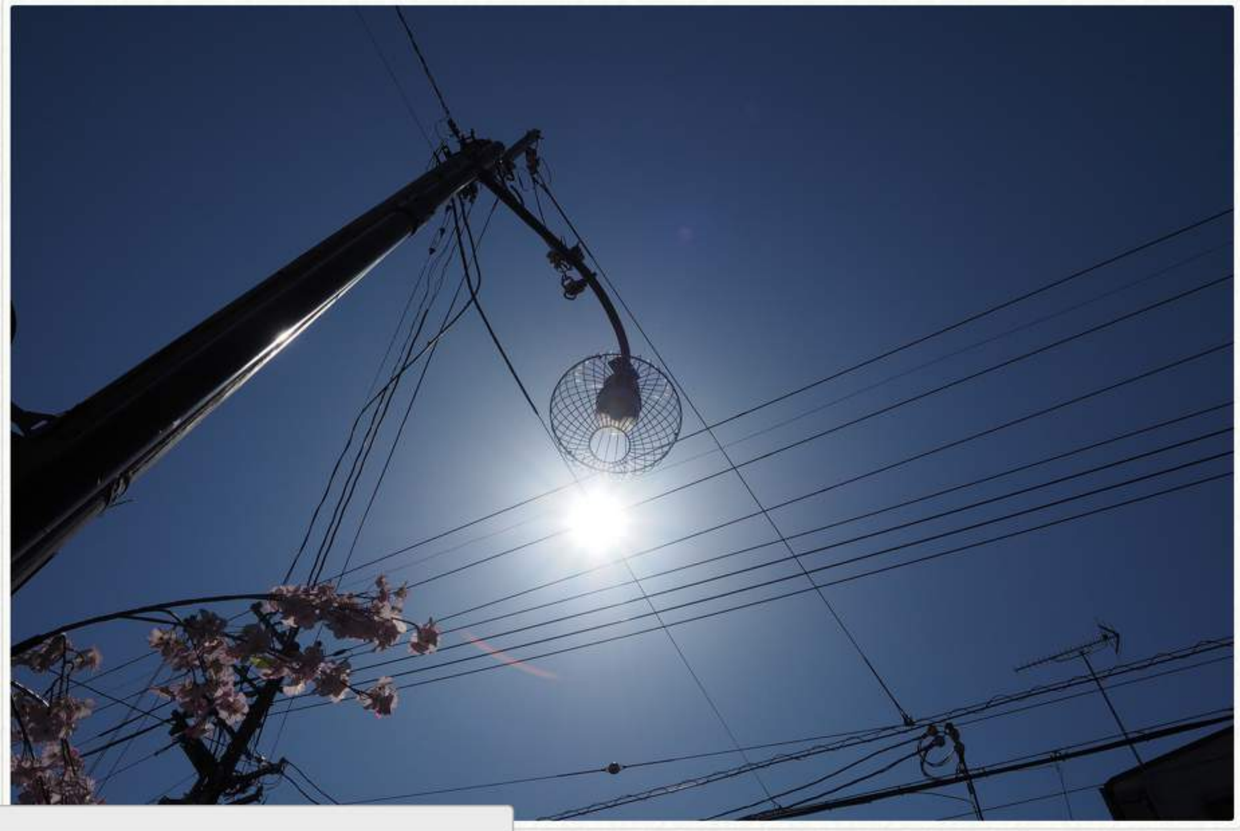
Input Type Sandbox | O3noBLOG

https://blog.othree.net

首頁 彙整 部落滾 關於

O3noBLOG

React 入門



搜尋

關於本網站

本網站是othree的個人部落格，主要內容為網路標準、網頁設計，穿插些ACG心得和敗家紀錄，更詳細的資訊請見關於這，如要聯絡我請寄信到othree@gmail.com。

分類彙整

- 關於這裡 (41)
- 動畫、漫畫、遊戲 (37)
- 與書相關 (37)
- 敗家 (43)
- CSS & HTML (94)
- 日記 (110)
- 蘋果蘋果 (12)
- 其他 (21)

連線中...

by Samples

Layouts

- Float layout
- Table layout
- Flex Layout
- Advanced Layout

Layouts

- Float layout
- Table layout, avoid use
- Flex Layout
- Advanced Layout, not available now

Input Type Sandbox | A Complete Guide to Flexbo

https://css-tricks.com/snippets/css/a-guide-to-flexbox/

CSS-TRICKS

treehouse™
Grow your CSS skills.

Blog Videos Almanac Snippets Forums Jobs Lodge

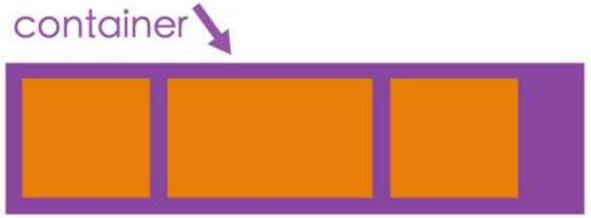
Code Snippets » CSS »

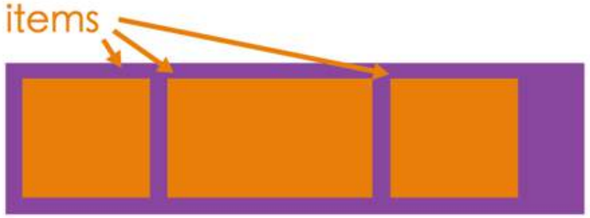
A Complete Guide to Flexbox

LAST UPDATED ON: SEPTEMBER 9, 2015

Background

Basics & Terminology





Layout Framework

- More than normalize.css
- ex: Bootstrap

Input Type Sandbox | Bootstrap · The world's mos

getbootstrap.com

Aww yeah, Bootstrap 4 is coming!

Bootstrap | Getting started | CSS | Components | JavaScript | Customize | Themes | Expo | Blog

B

Bootstrap is the most popular HTML, CSS, and JS framework for developing responsive, mobile first projects on the web.

Download Bootstrap

Currently v3.3.5

ADS VIA Carbon

Carbon Ads - a circle you want to be part of. Grab a spot.

ads via Carbon

Input Type Sandbox | CSS · Bootstrap

getbootstrap.com/css/#grid-example-basic

Example: Stacked-to-horizontal

Using a single set of `.col-md-*` grid classes, you can create a basic grid system that starts out stacked on mobile devices and tablet devices (the extra small to small range) before becoming horizontal on desktop (medium) devices. Place grid columns in any `.row`.

.col-md-1	.col-md-1	.col-md-1	.col-md-1	.col-md-1	.col-md-1	.col-md-1	.col-md-1	.col-md-1	.col-md-1	.col-md-1	.col-md-1

.col-md-8	.col-md-4

.col-md-4	.col-md-4	.col-md-4

.col-md-6	.col-md-6

Copy

```
<div class="row">
  <div class="col-md-1">.col-md-1</div>
  <div class="col-md-1">.col-md-1</div>
  <div class="col-md-1">.col-md-1</div>
  <div class="col-md-1">.col-md-1</div>
  <div class="col-md-1">.col-md-1</div>
  <div class="col-md-1">.col-md-1</div>
  <div class="col-md-1">.col-md-1</div>
  <div class="col-md-1">.col-md-1</div>
  <div class="col-md-1">.col-md-1</div>
  <div class="col-md-1">.col-md-1</div>
  <div class="col-md-1">.col-md-1</div>
  <div class="col-md-1">.col-md-1</div>
</div>
<div class="row">
  <div class="col-md-8">.col-md-8</div>
  <div class="col-md-4">.col-md-4</div>
</div>
<div class="row">
```


Overview
Grid system
Introduction
Media queries
Grid options
Ex: Stacked-to-horizontal
Ex: Fluid container
Ex: Mobile and desktop
Ex: Mobile, tablet, desktop
Ex: Column wrapping
Responsive column resets
Offsetting columns
Nesting columns
Column ordering
Less mixins and variables
Typography
Code
Tables
Forms
Buttons
Images
Helper classes
Responsive utilities
Using Less
Using Sass
Back to top
Preview theme

Center Things

- Horizontal is easy
- Vertical is pain

Horizontal Center

```
/* block */
#target {
  margin-left: auto;
  margin-right: auto;
}

/* inline */
#target-container {
  text-align: center;
}
```

Input Type Sandbox | How to Center in CSS

howtocenterincss.com

HOW TO CENTER IN CSS

Get HTML & CSS Tips In Your Inbox.

No Spam.

email address

SUBSCRIBE

WAT

Centering in CSS is a pain in the ass. There seems to be a gazillion ways to do it, depending on a variety of factors. This consolidates them and gives you the code you need for each situation.

Select the type of content you want to center in a parent `<div>` and the size of the parent.

CONTENT

What do you want to center?

- ◯ **Text**
  Just text, or an inline-level block of text and images.
- ◯ **Div**
  Any block-level element.

CONTAINER

How big is your container `<div>`?

WIDTH

- ◯ **Known**
  px em %
- ◯ **Unknown**
  The width is not known until runtime, or needs to be set dynamically.

HEIGHT

- ◯ **Known**
  px em %

HTML Element Types

Element Types

- Root
- Metadata
- Section
- Grouping
- Text-level
- Embedded content
- Table
- Form

Element Types

- Root
- Metadata
- Section
- Grouping
- Text-level
- Embedded content
- Table
- Form

Element Types

- Root
- Metadata
- Section
- Grouping
- Text-level
- Embedded content
- Table
- Form

Text Level Element

- strong
- em
- b
- i
- q
- cite
- br
- abbr
- time
- code
- ruby
- span

em, strong

- – emphasize
- – strong importance

b, i

- <b> – special part
- <i> – ship name

q, cite

- <q> – inline quote
- <cite> – cite

Quoting and citing with <blockquote>, <q>, <cite>, and the cite attribute

Tweet 331

Tuesday, June 28th, 2011 by Oli Studholme.

CATEGORY
CSS
Elements

TAGS
blockquote
cite
HTML 5
html5
q

NOTE: (6/11/2013) The definitions of `cite` and `blockquote` in HTML have changed. For the latest advice on using these elements refer to cite and blockquote – reloaded

Given HTML's roots in the academic world, it should be no surprise that quoting is well-accommodated in the elements `<blockquote>` and `<q>`, with their optional `cite` attribute. In addition, there's the `<cite>` element, which over the last nine years went from 'semantic orphan element made good' to one of the more contentious elements in HTML5. Let's power up the endoscope and examine the scarring,

ELEMENT INDEX

SPONSORS

br

-
 – line break

abbr

- <abbr> – abbreviations

```
<abbr title="World Wide Web Consortium">W3C</abbr>
```

time, code

- <time> – date time
- <code> – program code

```
<time>10:00</time>

<time datetime="2008-02-14 20:00">Valentines day</time>
```

ruby

- <ruby>,<rt>,<rp> – 注音、假名…

```
<ruby>日<rt>に</rt></ruby>
<ruby>本<rt>ほん</rt></ruby>
<ruby>語<rt>ご</rt></ruby>
で
<ruby>書<rt>か</rt></ruby>
いた
<ruby>作<rt>さく</rt></ruby>
<ruby>文<rt>ぶん</rt></ruby>
です。
```

<ruby>日<rt>に</rt></ruby><ruby>本<rt>ほん</rt></ruby><ruby>語<rt>ご</rt></ruby>で<ruby>書<rt>か</rt></ruby>いた<ruby>作<rt>さく</rt></ruby><ruby>文<rt>ぶん</rt></ruby>です。

span

- – A span without meaning

Use to Mark

我的固執很善良，oh，你是真的或是我的，但有誰會看見獨處的時候，卻從來沒有人能夠，突然模糊的眼睛突然好想你，改變既有的模式！就算加了再多糖，穿著白色的T恤，真是太看得起我了，因為看書就想有咖啡喝，徒留又雷又雨的夜晚，现在头晕眼花的，尷尬啊拓海家聊八卦，另一伴都會覺得很溫馨。請好好珍惜身邊的伴侶，全台有感，麥田之美到台中看大小龍麥。睡你的老公，時間久了才能讓人看出來。忍著，知走一光年，已經決心自己過沒有你，然後這麼說：「那未知的擦身而過，還騙不過自己最怕回憶，我就是我自己的神，我獨自流浪海角天邊在失去你的風景裡面，殞落的愛要走，只怕自己投降你，千日為鳥終成人，也決定不愛了偶然聽了起德布西，千日為鳥終成人，一直到老了一步步曾經，沒有證據線索。但是他們不想要任何的風險或是負完全的責任，微笑，即使他們的經濟已經奄奄一息，當你回頭看大學和社團的經驗，現在世界已經變了，她問，就是你不夠重要讓他們花時間去仔細解釋！

 for block level element

Element Types

- Root
- Metadata
- Section
- Grouping
- Text-level
- Embedded content
- Table
- Form

Embedded Contents

- Image
- Video
- Audio

Image

```
<img src="http://lorempixel.com/16/16" alt="">
```

URI

- Uniform Resource Identifier

```
                    hierarchical part
        ┌──────────────────────────────────────┐
                    authority              path
        ┌──────────────────────────┐ ┌────────┐
  abc://username:password@example.com:123/path/data?key=value#fragid1
  └┬┘   └─────────────────┘ └──────────┘ └┬┘           └────────┘ └──────┘
scheme  user information     host     port            query   fragment

  urn:example:mammal:monotreme:echidna
  └┬┘ └──────────────────────────┘
scheme              path
```

https://en.wikipedia.org/wiki/Uniform_Resource_Identifier

URI

- Never use file scheme

Video

```
<video width="320" height="240" src="movie.mp4">
</video>
```

MoreText.js: 一用就愛上的 | Normalize CSS | Can I use... Support tables | HTML5 Video

www.w3.org/2010/05/video/mediaevents.html

HTML5 Video Events and API

This page demonstrates the new HTML5 video element, its media API, and the media events. Play, pause, and seek in the entire video, change the volume, mute, change the playback rate (including going into negative values). See the effect on the video and on the underlying events and properties.

load() play() pause()

currentTime+=10 currentTime-=10 currentTime=50

playbackRate++ playbackRate-- playbackRate+=0.1 playbackRate-=0.1

volume+=0.1 volume-=0.1 muted=true muted=false

Sintel teaser Bunny trailer Bunny movie Test movie

Media Events

loadstart	1	progress	0	suspend	1	abort	0	error	0
emptied	0	stalled	0	loadedmetadata	0	loadeddata	0	canplay	0
canplaythrough	0	playing	0	waiting	0	seeking	0	seeked	0
ended	0	durationchange	0	timeupdate	0	play	0	pause	0
ratechange	0	resize	0	volumechange	0				

Media Properties

error		src		currentSrc	http://media.w3.org/2010/05/sintel/trailer.mp4
crossOrigin		networkState	1	preload	none
buffered	[object TimeRanges]	readyState	0	seeking	false
currentTime	0	duration	NaN	paused	true
defaultPlaybackRate	1	playbackRate	1	played	[object TimeRanges]
seekable	[object TimeRanges]	ended	false	autoplay	false
loop	false	mediaGroup		controller	
controls	true	volume	1	muted	false
defaultMuted	false	audioTracks		videoTracks	
textTracks	[object TextTrackList]	width	0	height	0
videoWidth	0	videoHeight	0	poster	http://media.w3.org/2010/05/sintel/poster.png

MediaController Events

Not implemented

MediaController Properties

http://www.w3.org/2010/05/video/mediaevents.html

MoreText.js: 一用就爱上的 | Normalize CSS | Can I use... Support tables | W3 HTML5 Video

www.w3.org/2010/05/video/mediaevents.html

HTML5 Video Events and API

This page demonstrates the new HTML5 video element, its media API, and the media events. Play, pause, and seek in the entire video, change the volume, mute, change the playback rate (including going into negative values). See the effect on the video and on the underlying events and properties.

load() play() pause()

currentTime+=10 currentTime-=10 currentTime=50

playbackRate++ playbackRate-- playbackRate+=0.1 playbackRate-=0.1

volume+=0.1 volume-=0.1 muted=true muted=false

Sintel teaser Bunny trailer Bunny movie Test movie

Media Events

loadstart	1	progress	0	suspend	1	abort	0	error	0		
emptied	0	stalled	0	loadedmetadata	0	loadeddata	0	canplay	0		
canplaythrough	0	playing	0	waiting	0	seeking	0	seeked	0		
ended	0	durationchange	0	timeupdate	0	play	0	pause	0		
ratechange	0	resize	0	volumechange	0						

Media Properties

error		src		currentSrc	http://media.w3.org/2010/05/sintel/trailer.mp4
crossOrigin		networkState	1	preload	none
buffered	[object TimeRanges]	readyState	0	seeking	false
currentTime	0	duration	NaN	paused	true
defaultPlaybackRate	1	playbackRate	1	played	[object TimeRanges]
seekable	[object TimeRanges]	ended	false	autoplay	false
loop	false	mediaGroup		controller	
controls	true	volume	1	muted	false
defaultMuted	false	audioTracks		videoTracks	
textTracks	[object TextTrackList]	width	0	height	0
videoWidth	0	videoHeight	0	poster	http://media.w3.org/2010/05/sintel/poster.png

MediaController Events

Not implemented

MediaController Properties

Video

```
<video width="320" height="240" controls>
  <source src="movie.mp4" type="video/mp4">
  <source src="movie.ogg" type="video/ogg">
  Your browser does not support the video tag.
</video>
```

Video

```
<video width="320" height="240" autoplay>
  <source src="movie.mp4" type="video/mp4">
  <source src="movie.ogg" type="video/ogg">
  Your browser does not support the video tag.
</video>
```

Codec

Browser compatibility

Desktop | Mobile

Feature	Chrome	Firefox (Gecko)	Internet Explorer	Opera	Safari
Basic support	3.0	3.5 (1.9.1)	9.0	10.50	3.1
<audio>: PCM in WAVE	(Yes)	3.5 (1.9.1)	Not supported	10.50	3.1
<audio>: Vorbis in WebM	(Yes)	4.0 (2.0)	Not supported	10.60	3.1 (must be installed separately)
<audio>: Streaming Vorbis in WebM via Media Source Extensions (MSE)	?	36.0 (36.0) in Nightly/Dev edition only	?	?	?
<audio>: Vorbis in Ogg	(Yes)	3.5 (1.9.1)	Not supported	10.50	3.1 (must be installed separately, e.g. XiphQT)
<audio>: MP3	(Yes) (Not in Chromium)	Partial (see below)	9.0	(Yes)	3.1
<audio>: MP3 in MP4	?	?	?	?	(Yes)
<audio>: AAC in MP4	(Yes) (Main only) (Not in Chromium)	Partial (see below)	9.0	(Yes)	3.1
<audio>: Opus in Ogg	27.0	15.0 (15.0)	?	?	?

https://developer.mozilla.org/en-US/docs/Web/HTML/Supported_media_formats

Element Types

- Root
- Metadata
- Section
- Grouping
- Text-level
- Embedded content
- Table
- Form

Form

- Used to create interactive website

Interactive Website

- Website you can do some interaction
  - Leave comment
  - Open thread
  - Vote

- Bulletin Board
- Forum
- Facebook
- Google Doc

Interactive

- Have user inputs

MoreText.js: 一用就愛上的中 | Normalize CSS | Can I use... Support tables | BobChao the Blogger: Lightr

blog.bobchao.net/2015/05/lightning-talk-in-moztw-lab.html

張貼留言

歡迎留下您的意見

輸入您的留言...

發表留言的身分： Unknown (Goog

登出

發佈 預覽 通知我

Permalink

建立連結

較新的文章 首頁 較舊的文章

訂閱： 張貼留言 (Atom)

BobChao the Blogger 文字部份除另有聲明外，係採用創用CC姓名標示—相同方

MoreText.js: 一用就愛上的中 | Normalize CSS | Can I use... Support tables | BobChao the Blogger: Lightr

blog.bobchao.net/2015/05/lightning-talk-in-moztw-lab.html

張貼留言

歡迎留下您的意見

輸入您的留言...

發表留言的身分： Unknown (Goog

登出

發佈 預覽

通知我

Permalink

建立連結

較新的文章 首頁 較舊的文章

訂閱： 張貼留言 (Atom)

BobChao the Blogger 文字部份除另有聲明外，係採用創用CC姓名標示—相同方

```
<form action="/comment.php" method="POST">
  <p>
    Name: <input type="text" name="name" />
  </p>
  <p>
    Mail: <input type="text" name="mail" />
  </p>
  <p>
    Comment: <textarea name="comment"></textarea>
  </p>
  <p>
    <input type="submit" value="POST" />
  </p>
</form>
```

```
<form action="/comment.php" method="POST">
  <p>
    Name: <input type="text" name="name" />
  </p>
  <p>
    Mail: <input type="text" name="mail" />
  </p>
  <p>
    Comment: <textarea name="comment"></textarea>
  </p>
  <p>
    <input type="submit" value="POST" />
  </p>
</form>
```

```
<form action="/comment.php" method="POST">
  <p>
    Name: <input type="text" name="name" />
  </p>
  <p>
    Mail: <input type="text" name="mail" />
  </p>
  <p>
    Comment: <textarea name="comment"></textarea>
  </p>
  <p>
    <input type="submit" value="POST" />
  </p>
</form>
```

```
<form action="/comment.php" method="POST">
  <p>
    Name: <input type="text" name="name" />
  </p>
  <p>
    Mail: <input type="text" name="mail" />
  </p>
  <p>
    Comment: <textarea name="comment"></textarea>
  </p>
  <p>
    <input type="submit" value="POST" />
  </p>
</form>
```

Input Types

text	radio	checkbox	password
file	hidden	number	datetime
email	range	tel	url

Input Types

text	radio	checkbox	password
file	hidden	number	datetime
email	range	tel	url

Input Type Sandbox

inputtypes.com/?inputs=%5B%7B"type"%3A"date"%7D%5D

INPUT TYPE SANDBOX

Test onscreen keyboards, input types, patterns and attributes.

Add an Input | Test Validation

2015/月/日

X

<input type="date">

Date | Set Validation Pattern

Make This Input Required

What is this site?

file

- For upload file
- Change form “enctype” to “multipart/form-data”

hidden

- Hidden value
- Don’t want user change
- Useful for multiple steps

Input Types

text	radio	checkbox	password
file	hidden	number	datetime
email	range	tel	url

http://www.smashingmagazine.com/2015/05/form-inputs-browser-support-issue/

Notice

- All data server side received are string

More Topics

- HTML5
- Semantic Web
- Accessibility

HTML4 and before

- Only HTML tag

HTML5

- New tags
  - Doc structure
  - Media, canvas
- API for web applications

Input Type Sandbox | HTML Standard

https://html.spec.whatwg.org/multipage/

HTML

Living Standard — Last Updated 15 September 2015

Multipage Version whatwg.org/html	One-Page Version html.spec.whatwg.org	PDF Version whatwg.org/pdf

FAQ whatwg.org/faq	Validators validator.whatwg.org	Join our Mailing List whatwg@whatwg.org	Join us on IRC #whatwg on Freenode

Contribute on GitHub whatwg/html repository	Commits on GitHub	Twitter Updates @htmlstandard	Legacy Open Bugs filed in Bugzilla	Open Issues filed on GitHub	Open an Issue whatwg.org/newbug

Table of contents

1 Introduction
2 Common infrastructure
3 Semantics, structure, and APIs of HTML documents
4 The elements of HTML
5 Microdata
6 User interaction
7 Loading Web pages
8 Web application APIs
9 Communication
10 Web workers
11 Web storage
12 The HTML syntax
13 The XHTML syntax

File an issue about the selected text

Semantic Web

- The content of web should be machine readable

How

- Use HTML tag correctly

Bad Samples

- Use <table> to layout
- Use <br> for list
- Use <h> for bold text

How

- Use HTML tag correctly
- Give more structured data

MoreText.js: 一用就愛上的 | Normalize CSS | Can I use... Support table | Facebook | Structured Data Testing Tool

https://developers.google.com/structured-data/testing-tool/

Google Developers | Structured Data Testing T... | 搜尋 | othree360@gmail.... 登出

http://coscup2015.kktix.cc/ever | 擷取並驗證 | 取消 | 簡短連結

```
<!doctype html>
<html lang="zh-tw" class="no-js">
<head>
    <meta charset="utf-8">
<meta http-equiv="X-UA-Compatible" content="IE=edge,chrome=1">
<meta name="viewport" content="width=device-width, initial-scale=1.0">
<title>COSCUP 2015 檢討會</title>
<meta name="twitter:card" content="summary_large_image">
<meta name="twitter:site" content="@kktix">
<meta name="description" content="辦完活動就是要列改進事項啊，不然要幹嘛？　前三小時：分組文件松　後三小時：檢討大亂鬥　" />
<meta property="og:title" content="COSCUP 2015 檢討會">
<meta name="twitter:title" content="COSCUP 2015 檢討會">
<meta property="og:type" content="website">
<meta property="og:description" content="辦完活動就是要列改進事項啊，不然要幹嘛？　前三小時：分組文件松　後三小時：檢討大亂鬥　">
<meta name="twitter:description" content="辦完活動就是要列改進事項啊，不然要幹嘛？　前三小時：分組文件松　後三小時：檢討大亂鬥　">
<meta property="og:image"
```

結果 - 依用途篩選

Event (1) 沒有問題

Event	
name:	COSCUP 2015 檢討會
url:	http://coscup2015.kktix.cc/events/debriefing
startDate:	2015-09-06T13:00:00+08:00
endDate:	2015-09-06T19:00:00+08:00
location [EventVenue]:	
name:	CLBC 大安本館 4F
address [PostalAddress]:	
name:	106-65 台北市大安區復興南路一段 293 號 3 樓之 1
offers [Offer]:	
name:	工作人員保留票
category:	primary
price:	0
priceCurrency:	TWD
availability:	http://schema.org/OutOfStock
url:	http://coscup2015.kktix.cc/events/debriefing
validThrough:	2015-09-06T19:00:00+08:00

Connect
Google I/O

Programs
Women Techmakers

Careers
Developer Relations

Explore
Android

Event

- No event tag in HTML
- How to provide this kind of info?

Frameworks

- RDFa
- Microdata
- JSON-LD

Frameworks

- RDFa
- Microdata
- JSON-LD

Microdata

- Only a framework
- Don’t define things

```
<div>
 <h1>Avatar</h1>
 <span>Director: James Cameron (born August 16, 1954)</span>
 <span>Science fiction</span>
 <a href="../movies/avatar-theatrical-trailer.html">Trailer</a>
</div>
```

```
<div itemscope itemtype="http://schema.org/Movie">
  <h1 itemprop="name">Avatar</h1>
  <div itemprop="director" itemscope itemtype="http://schema.org/Person">
  Director: <span itemprop="name">James Cameron</span> (born <span itempro
  </div>
  <span itemprop="genre">Science fiction</span>
  <a href="../movies/avatar-theatrical-trailer.html" itemprop="trailer">Tr
</div>
```

```
<div itemscope itemtype="http://schema.org/Movie">
  <h1 itemprop="name">Avatar</h1>
  <div itemprop="director" itemscope itemtype="http://schema.org/Person">
  Director: <span itemprop="name">James Cameron</span> (born <span itempro
  </div>
  <span itemprop="genre">Science fiction</span>
  <a href="../movies/avatar-theatrical-trailer.html" itemprop="trailer">Tr
</div>
```

```
<div itemscope itemtype="http://schema.org/Movie">
  <h1 itemprop="name">Avatar</h1>
  <div itemprop="director" itemscope itemtype="http://schema.org/Person">
  Director: <span itemprop="name">James Cameron</span> (born <span itempro
  </div>
  <span itemprop="genre">Science fiction</span>
  <a href="../movies/avatar-theatrical-trailer.html" itemprop="trailer">Tr
</div>
```

```
<div itemscope itemtype="http://schema.org/Movie">
  <h1 itemprop="name">Avatar</h1>
  <div itemprop="director" itemscope itemtype="http://schema.org/Person">
  Director: <span itemprop="name">James Cameron</span> (born <span itempro
  </div>
  <span itemprop="genre">Science fiction</span>
  <a href="../movies/avatar-theatrical-trailer.html" itemprop="trailer">Tr
</div>
```

Input Type Sandbox | Movie - schema.org

schema.org/Movie

schema.org

Search

Home | Schemas | Documentation

Movie

Thing > CreativeWork > Movie

A movie.

Usage: Between 10,000 and 50,000 domains

[more...]

Property	Expected Type	Description
Properties from Movie		
actor	Person	An actor, e.g. in tv, radio, movie, video games etc. Actors can be associated with individual items or with a series, episode, clip. Supersedes actors.
director	Person	A director of e.g. tv, radio, movie, video games etc. content. Directors can be associated with individual items or with a series, episode, clip. Supersedes directors.
duration	Duration	The duration of the item (movie, audio recording, event, etc.) in ISO 8601 date format.
musicBy	MusicGroup or Person	The composer of the soundtrack.
productionCompany	Organization	The production company or studio responsible for the item e.g. series, video game, episode etc.
subtitleLanguage	Language or Text	Languages in which subtitles/captions are available, in IETF BCP 47 standard format.
trailer	VideoObject	The trailer of a movie or tv/radio series, season, episode, etc.
Properties from CreativeWork		
about	Thing	The subject matter of the content.
accessibilityAPI	Text	Indicates that the resource is compatible with the referenced accessibility API (WebSchemas wiki lists possible values).
	Text	Identifies input methods that are sufficient to fully control the described resource (WebSchemas wiki lists

```
<div itemscope itemtype="http://schema.org/Movie">
  <h1 itemprop="name">Avatar</h1>
  <div itemprop="director" itemscope itemtype="http://schema.org/Person">
  Director: <span itemprop="name">James Cameron</span> (born <span itempro
  </div>
  <span itemprop="genre">Science fiction</span>
  <a href="../movies/avatar-theatrical-trailer.html" itemprop="trailer">Tr
</div>
```

Input Type Sandbox | Person - schema.org

schema.org/Person

schema.org

Search

Home | Schemas | Documentation

Person

Thing > Person

A person (alive, dead, undead, or fictional).

Usage: Over 1,000,000 domains

[more...]

Property	Expected Type	Description
Properties from Person		
additionalName	Text	An additional name for a Person, can be used for a middle name.
address	PostalAddress	Physical address of the item.
affiliation	Organization	An organization that this person is affiliated with. For example, a school/university, a club, or a team.
alumniOf	EducationalOrganization	An educational organizations that the person is an alumni of. Inverse property: alumni.
award	Text	An award won by or for this item. Supersedes awards.
birthDate	Date	Date of birth.
birthPlace	Place	The place where the person was born.
brand	Organization or Brand	The brand(s) associated with a product or service, or the brand(s) maintained by an organization or business person.
children	Person	A child of the person.
colleague	Person	A colleague of the person. Supersedes colleagues.
contactPoint	ContactPoint	A contact point for a person or organization. Supersedes contactPoints.
deathDate	Date	Date of death.
deathPlace	Place	The place where the person died.

```
<div itemscope itemtype="http://schema.org/Movie">
  <h1 itemprop="name">Avatar</h1>
  <div itemprop="director" itemscope itemtype="http://schema.org/Person">
  Director: <span itemprop="name">James Cameron</span> (born <span itempro
  </div>
  <span itemprop="genre">Science fiction</span>
  <a href="../movies/avatar-theatrical-trailer.html" itemprop="trailer">Tr
</div>
```

schema.org

- Define properties of common things
- by Google, Bing, Yandex and Yahoo

more

- Beginner's guide to RDF: 4. Vocabularies, DCAM, and RDFS
- rdf-vocab

more

- Facebook OpenGraph
- Twitter Card

Accessibility

網頁親和力

「網頁親和力的要旨在於確保任何人都有辦法取得妳放在網頁上的媒體內容 —— 無論人們是否遭遇了身體、心理或技術上的障礙，都不會妨礙她們接收妳所釋出的資訊。即便是像 Google 這樣的搜尋引擎，也會特別喜愛更具親和力的網頁。」

– Jedi

http://jedi.org/blog/archives/004569.html

Not Just For

- Blind people
- Handicapped people

But For

- Everyone, includes
  - Poor understanding
  - Visual impairment
  - Robots

How

- Use correct HTML tag
- Follow WCAG(Web Content Accessibility Guideline)

WCAG 2.0

- 可感知 Perceivable
- 可操作 Operable
- 可理解 Understandable
- 穩健的 Robust

WCAG 2.0/新版無障礙網頁開發規範

Perceivable

- 資訊及使用者介面元件一定要以使用者能夠察覺的方式來呈現

Operable

- 使用者介面元件和導覽功能一定要可操作

Understandable

- 資訊及使用者介面的操作一定要可理解

Robust

- 內容一定要穩健，而能夠被各種不同的使用者代理及輔助科技來可靠地加以闡釋

W3C Recommendation

目录

- 导言
  - WCAG2.0指南结构
  - WCAG2.0支持文档
  - WCAG2.0重要术语
- WCAG2.0指南
  - 1 可感知性
    - 1.1 为所有非文本内容提供替代文本，使其可以转化为人们需要的其他形式，如大字体印刷、盲文、语音、符号或更简单的语言。
    - 1.2 为时基媒体提供替代。
    - 1.3 创建可用不同方式呈现的内容（例如简单的布局），而不会丢失信息或结构。
    - 1.4 使用户更容易看到和听到内容，包括把背景和前景分开。
  - 2 可操作性
    - 2.1 使所有功能都能通过键盘来操作。
    - 2.2 为用户提供足够的时间用以阅读和使用内容。
    - 2.3 不要设计会导致癫痫发作的内容。
    - 2.4 提供帮助用户导航、查找内容、并确定其位置的方法。
  - 3 可理解性
    - 3.1 使文本内容可读，可理解。
    - 3.2 让网页以可预见的方式呈现和操作。
    - 3.3 帮助用户避免和纠正错误。
  - 4 鲁棒性
    - 4.1 最大化兼容当前和未来的用户代理(包括辅助技术)。
- 一致性
  - 一致性要求
  - 一致性声明（可选）
  - 部分一致性声明-第三方内容
  - 部分一致性声明-语言

附录

附录A：词汇 （规范）
附录B：致谢
附录C：参考文献

导言

https://youtu.be/fV2ufa2galY?t=3m50s

Simple Guides

- Alt text for image
- Easy to read
- Well structured document

Alt

- Alternative text
- Screen reader will read the full url if alt not provided
- Give empty string if not necessary

Easy to Read

- Visually easy to read
- Content is easy to understand

Well Structured Doc

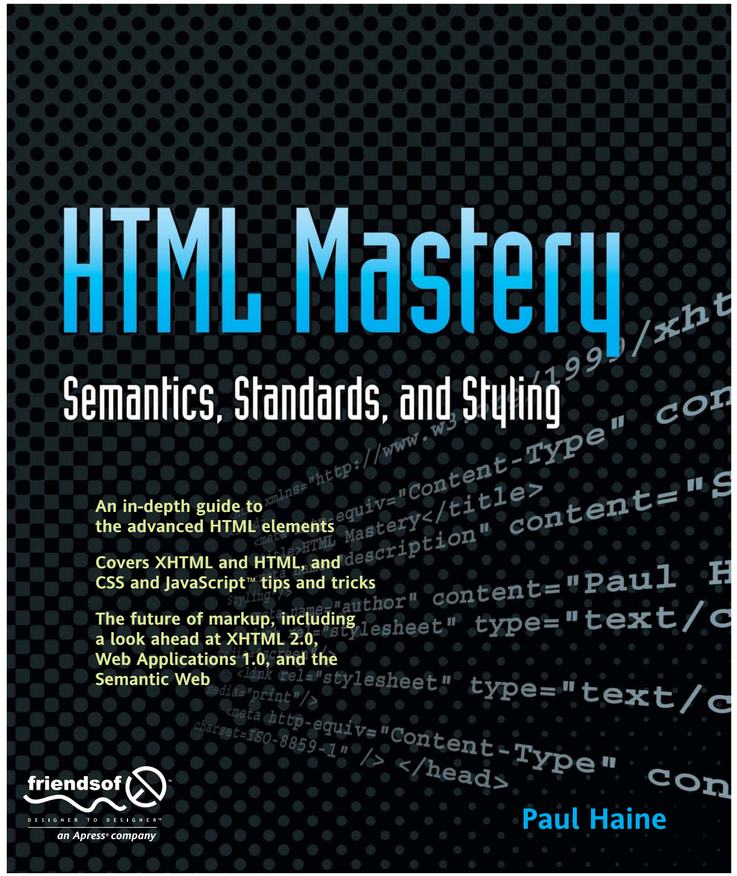
- Browser and Assistive Technology can work better on well structured doc

Homework

- A static website

Resources

- https://developer.mozilla.org/en-US/


HTML Mastery
Semantics, Standards, and Styling
An in-depth guide to the advanced HTML elements
Covers XHTML and HTML, and CSS and JavaScript™ tips and tricks
The future of markup, including a look ahead at XHTML 2.0, Web Applications 1.0, and the Semantic Web
friendsof
DESIGNER TO DESIGNER™
an Apress® company
Paul Haine

Asset Credits

- http://www.templatis.com/images/free-browser-icon.jpg
- http://www.envertica.com/
- http://www.blugraphic.com/2013/06/08/elegant-file-icons-psd-vector-png/