Web前端基础

《Web应用开发》 / 任课教师：罗志一

计算机科学与技术学院

浙江理工大学
ZHEJIANG SCI-TECH UNIVERSITY

概览

- HTML：创建超媒体文档结构和内容
- CSS：控制视觉效果
- Javascript：人机交互的工具语言
- 正则表达式：网页解析

如何部署

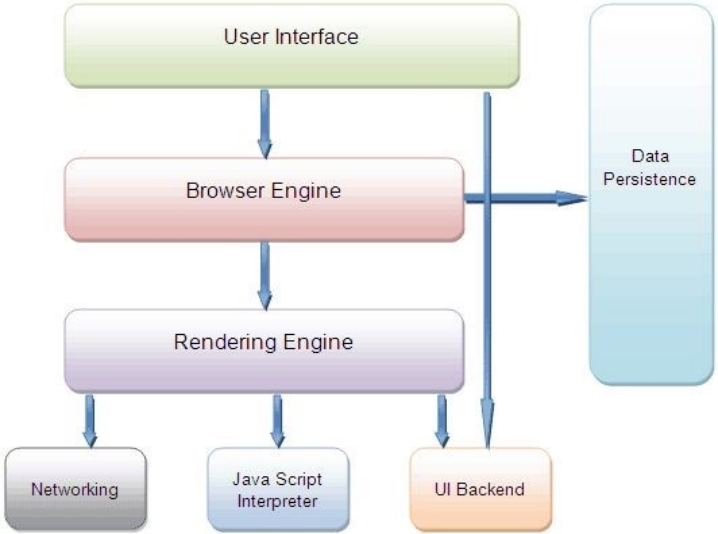
- 开始准备工作
  - 网页浏览器（Chrome或者Firefox）
  - 示例HTML代码（Download）
  - 好的文本编辑器：Editplus（win），VSCode（跨平台），textWrangler（osx），vim（unix）以及sublime text（跨平台）

代码测试

- 在文本编辑器和浏览器中打开template文件夹中的index.html文件（准备工作中示例代码下载）
- 当对代码修改后，在浏览器中按F5（刷新）检查网页变化
- 打开开发者功能，快捷键：
  - Windows：Control+Shift+I 或者 F12
  - OSX：Command+Opt+I
  - 其他在线编辑器：scratchpad 或者 htmledit

浏览器剖析

- 浏览器内部有很多不同的部分，我们主要学习下面两个部分
  - 页面渲染引擎：负责将我们的 HTML + CSS 转换成可视化图像
  - JavaScript解释器（VM）：负责执行Javascript代码



Web前端技术

- HTML
- CSS
- Javascript

HTML

- Hyper Text Makeup Language(HTML):超文本标记文件
  - 定义文档或者网站的结构
  - 是标记语言而不是编程语言，目的是为网站的内容提供结构而不是定义一种算法
  - 包含网站所有信息的一系列嵌套标记，eg：
    `<title>This is a title</title>`
  - HTML定义页的结构，一个网站可以有不同的HTML对应不同的页面

```
<html>
    <head>
    </head>
    <body>
        <div>
            <p>Hi</p>
        </div>
    </body>
</html>
```

HTML的基础规则

- 使用XML语法（带有属性的标记，可以包含其他标记），例如：

```
<tag_name attribute="value"> content </tag_name>
```

- **存储**必须显示给用户的所有信息
- 对于不同类型的信息和行为，有不同的HTML元素
- 信息需要存储在一个被叫做文档对象模型（DOM）的树形结构（包含内部节点的节点）中
- 要给出文档的语义结构，有助于计算机了解网站内容
- 不包含应该如何显示的相关信息(该信息属于 CSS) ，因此没有颜色信息、字体大小、位置等。

语法例子

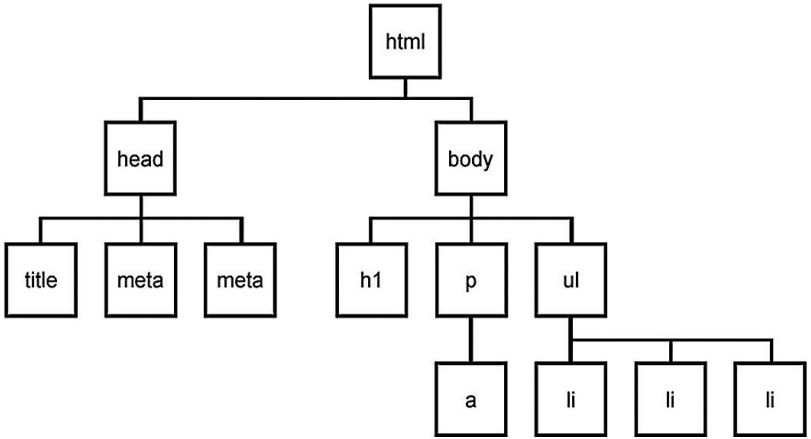

标签名
属性

注释
<!-- this is a comment -->
文本标记
This is text without a tag.
<button class="mini">press me</button>

自关闭标签
(self-closing tag)

文档对象模型(DOM)

- 文档对象模型为树结构：每个节点只能有一个父节点，每个节点可以有几个子节点，所以结构看起来像一棵树。



文档对象模型(DOM)


Article
Talk
WIKIPEDIA
The Free Encyclopedia
Main page
Contents
Featured content
Current events
Random article
Donate to Wikipedia
Interaction
Help
About Wikipedia
Community portal
Recent changes
Contact Wikipedia
Toolbox
Print/export
Languages
Afrikaans
Alemannisch
Asturianu
Azərbaycanca
Bân-lâm-gú
Беларуская
Беларуская (тарашкевіца)
Български
Bosanski
Brezhoneg
Català
Чăвашла
Česky
Firefox
From Wikipedia, the free encyclopedia
For other uses, see Firefox (disambiguation)
"Phoenix (web browser)" redirects here. For the Phoenix browser based on tkWWW, see tkWWW
Mozilla Firefox is a free and open source web browser descended from the Mozilla Application Suite and managed by Mozilla Corporation.
As of January 2012, Firefox has approximately 25% worldwide usage share of web browsers, making it the second or the third most widely used browser, according to different estimates. The browser has had particular success in Germany and France, where it is the most popular browser with 52% usage and 45% respectively.
To display web pages, Firefox uses the Gecko layout engine which implements most current web standards in addition to several features that are intended to anticipate likely additions to the standards.
Firefox runs on various operating systems including Microsoft Windows, Linux, Mac OS X, FreeBSD and many other platforms. Its current stable release is version 11.0, released on March 13, 2012.
Contents [hide]
1 History
1.1 Early versions
1.2 Version 2.0
1.3 Version 3.0
1.4 Version 3.5
1.5 Version 3.6
1.5.1 Out-of-process plug-ins
1.6 Version 4.0
1.7 Rapid release
1.7.1 Version 5.0
1.7.2 Version 6.0
1.7.3 Version 7.0
1.7.4 Version 8.0
1.7.5 Version 9.0
1.7.6 Version 10.0
1.7.7 Version 11.0
1.8 Future releases
1.8.1 Version 12.0
1.8.2 Version 13.0
2 Version
2.1 Released versions
2.2 Development versions table
Release (ESR)
mozilla
Firefox

主要的HTML标签

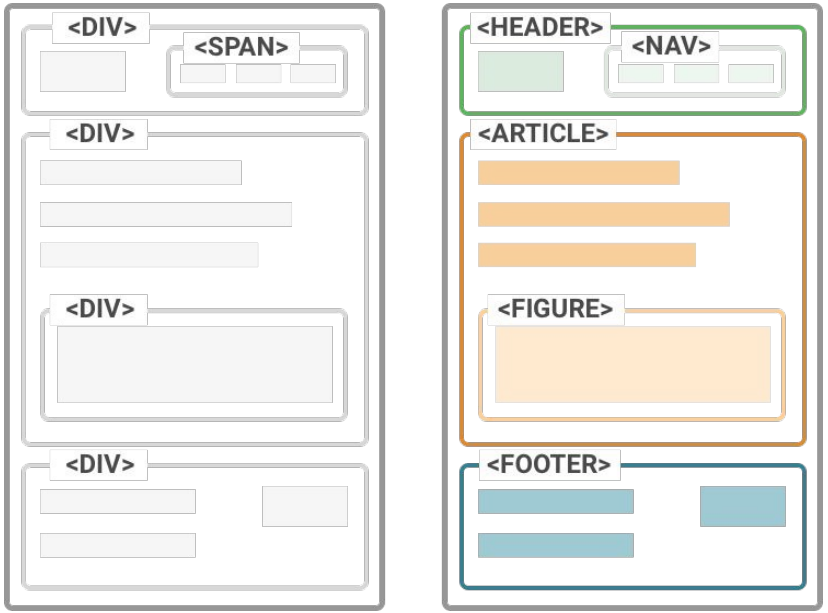
- 尽管HTML规范中有很多标签，但是99%的网站使用少于10个标签的HTML标签子集，其中最重要的是：
  - :一个容器，通常表示一个矩形区域，里面有信息
  - <img/>:图片
  - <a>:指向另一个URL的可点击链接
  - <p>:文本段落
  - <h1>:标题(h2，h3，h4是重要性递减的标题)
  - <input>:允许用户介绍信息的小部件
  - <style>:插入CSS规则
  - <script>:执行Javascript
  - :空标签

其他同样重要的标签

- <button>:创建一个按钮（button）
- <audio>:用于播放音频
- <video>:用于播放视频
- <canvas>:从 javascript 中绘制图形
- <iframe>:将另一个网站放进我们的网站

信息包装

我们使用 HTML 标签来包装网站上的不同信息。信息的结构越多，就越容易访问和呈现它。我们可以根据信息所在的标签改变信息在屏幕上的表示方式，所以我们不必担心使用太多的标签。



正确的标记方式

需要去避免的标记方式：

```
<div>
Title
Here is some content
</div>
```

DONT DO THIS

正确的标记方式：

```
<div>
      <h1>Title<h1>
      <p>Here is sone content</p>
</div>
```

合理使用HTML

- 最好将所有信息正确地包装在标记中，使其具有一定的语义，可以通过向标记添加额外的属性来扩展代码语义：
  - `id`：标签的唯一标识符
  - `class`：标签的通用标识符

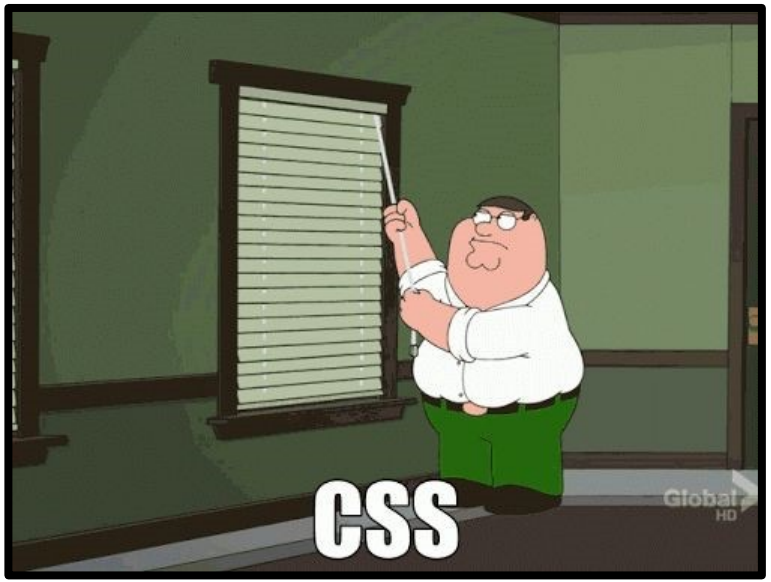
```
<div id="profile-picture" class="mini-image">...</div>
```

参考文档

- HTML Reference：所有HTML标签的描述
- The 25 Most used tags：最常用的标签信息
- HTML5 Good practices：一些建议

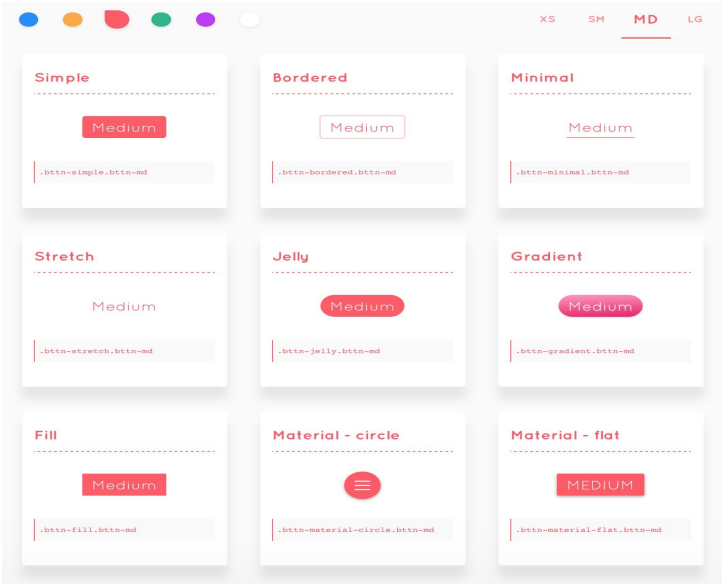
Web技术

- HTML
- CSS
- Javascript



CSS

- CSS 允许我们指定如何呈现(渲染)存储在HTML中的文档信息，利用CSS，我们可以控制可视化的所有方面和其他一些特性：
  - 颜色：内容、背景、边界
  - 边距：内部边距、外部边距
  - 位置：控制放在哪里
  - 尺寸：长度、宽度
  - 行为：鼠标上的变化



CSS例子

```
* {
    color: blue;   /*a comment */
    margin: 10px;
    font: 14px Tahoma;
}
```

以上代码改变网页里的所有标签(’*’意味着所有)显示为14px大小的蓝色Tahoma字体，并留下一个10px 左右的边距。

CSS字段

- 常用的CSS字段和例子
  - color： #FF0000; red; rgba(255,00,100,1.0); //指定颜色的不同方法
  - background-color: red;
  - background-image: url('file.png’);
  - font: 18px 'Tahoma’;
  - border: 2px solid black;
  - border-top: 2px solid red;
  - border-radius: 2px; //去掉角，使它们更圆
  - margin: 10px; //从边界到外部元素的距离
  - padding: 2px; //从边界到内部元素的距离
  - width: 100%; 300px; 1.3em; //许多不同的指定距离的方法
  - height: 200px;
  - text-align: center;
  - box-shadow: 3px 3px 5px black;
  - cursor: pointer;
  - display: inline-block;
  - overflow: hidden;

添加CSS规则

- 在样式标记中插入代码

```
<style>
        p {color : blue}
</style>
```

- 引用外部CSS文件

```
<link href="style.css" rel="stylesheet" />
```

- 在标记上使用属性样式

```
<p style="color: blue; margin: 10px">
```

- 使用Javascript

CSS选择器

让我们先从改变网站标签的背景颜色开始：

```
div  {
        background-color : red
}
```

这个 CSS 规则意味着在我们的网站中找到的每个标签 DIV 都应该有一个红色的背景色。DIV 主要用于表示我们网站的区域。

我们也可以通过影响标签主体来改变整个网站的背景:

```
body  {
        background-color : red
}
```

CSS选择器

如果我们想要更改一个特定的标记(不是同一类型的所有标记)，该怎么办？

除了标记的名称之外，我们还可以指定更精确的选择器。例如，通过类名称或id名。要指定具有给定类名的标记，我们使用点（dot）：

```
p.intro {
        color : red
}
```

这将只影响带有类名称 intro 的标记 p:

```
<p class=“intro”>
```

CSS选择器

- 主要的选择器
  - 标签名称：只是标签的名称
    - `p {...}  // 影响到所有<p>标记`
  - 点 **.**：影响某类的标记
    - `p.highlight {...} //影响类为highlight的<p>标记`
  - 锐化字符 **#**：指定带有该id的标记
    - `p#intro {...} //影响id为intro的<p>标记`
  - 冒号 **:**：行为状态(鼠标在上)
    - `p:hover {...} //影响鼠标所选的<p>标记`
  - 方括号 **[attr =‘value’]**：带有属性 attr 的标记，其值为‘ value’
    - `input[type=“text”] {...} //影响到文本类型的输入标记`

CSS选择器

还可以根据上下文指定标记，例如: 标记内部的标记与选择器匹配。只需用一个空格分隔选择器:

```
div#main p.intro { ... }
```

这个将影响id为main的div标签中class为intro的p标签

```
<div id="main">
        <p class=“intro”>....</p>  ← 影响这一个
</div>

<p class=“intro”>....</p>  ← 但是不影响这个
```

CSS选择器

可以组合选择器去更精确地选择某个标签

```
div#main.intro:hover { ... }
```

如果鼠标浮动，将会将CSS应用到所有id为main，class为intro的div标签

其实不需要指定一个标签，你可以使用类或者不带标签的 id 选择器，这意味着它会影响到 id为main 的任何节点

```
#main { ... }
```

CSS选择器

如果只想选择一个元素的直接子元素，使用 > 字符：

```
ul.menu > li { ... }
```

最后，如果你想对几个选择器使用相同的 CSS 操作，你可以使用逗号，字符：

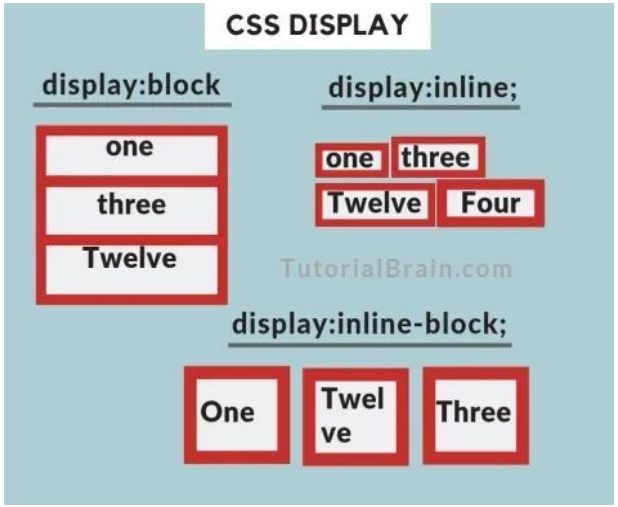
```
div, p { ... }  ← 这个将作用于所有div和p标签
```

HTML 格式

理解浏览器如何安排屏幕上的元素是很重要的，可以查阅教程，该教程解释了一个元素在屏幕上不同的排列方式，可以使用 display 属性更改元素的排列方式:

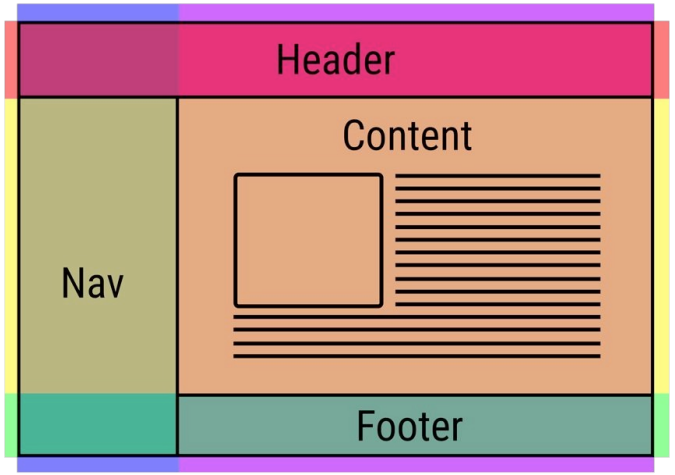
```
div { display: inline-block; }
```

同时还要需要检查属性`float`



布局

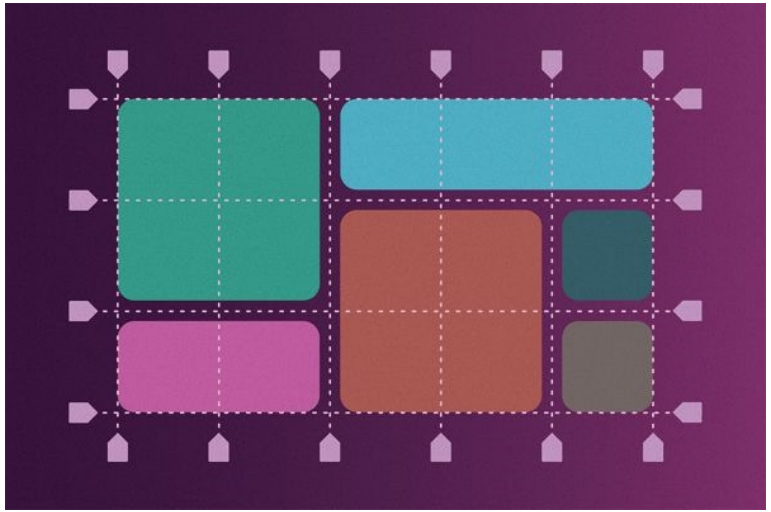
- CSS 最难的部分之一是构建网站的布局(窗口内的结构)。
- 默认情况下，HTML倾向于将所有内容放在一列中，这并不理想
- CSS 中已经有很多解决这个问题的方案(表、固定 div、 flex、 grid、 …)



网格系统

- 由于大多数网站都是网格结构，这里推荐使用 CSS 网格系统

- 查询该教程以轻松创建站点结构



HTML

```html
<div class="grid-container">
  <div class="grid-item1">1</div>
  <div class="grid-item2">2</div>
</div>
```

CSS

```css
.grid-container {
  display: grid;
  grid-template-rows: 100px; 100px;
  grid-template-columns: 100px; 100px; 100px;
  grid-gap: 5px;
}

.grid-item1 {
  background: blue;
  border: black 5px solid;
  grid-column-start: 1;
  grid-column-end: 5;
  grid-row-start: 1;
  grid-row-end: 3;
}
```

全屏幕divs

- 有时候我们想要一个覆盖整个屏幕的 `div`（用来制作网络应用程序），而不是一个滚动的网站(更像是普通的文档)

- 在这种情况下，要使用百分比来定义元素的大小，因为百分比是相对于父元素的大小，所以需要将百分比设置为100%

```
CSS

html, body {
      width: 100%;
      height: 100%;
}

div {
      margin: 0;
      padding: 0;
}

#main {
      width: 100%;
      height: 100%;
}
```

居中

有时候使divs居中比较困难，需要使用下面的技巧：

```
.horizontal-and-vertical-centering{
        display: flex;
        justify- content: center;
        align-items: center;
}
```

有关CSS的进一步阅读

对于CSS选择器而言还有更多的规则，可以查阅以下的链接去更好地理解这些规则：

- One line layouts tutorials
- Understanding the Box Model ：关于如何在你的文档上放置信息
- All CSS Selectors ： CSS 选择器规范页
- CSS Transition :如何使用 CSS 制作动画

Web技术

- HTML
- CSS
- Javascript

Javascript特点

- 一种规范的编程语言，易于入门，难于掌握
- 给网页上的元素增加联系，增强网页的交互性
- 语法类似于 C 或 Java，但没有类型
- 可以更改应用于元素的 HTML 或 CSS 的内容。
- 甚至可以从互联网上发送或检索信息来更新网页内容，而无需重新加载页面。

代码插入

- 使用 <script> 标记将代码嵌入HTML中

```
<script> /* some code */ </script>
```

- 使用 <script>标记导入Javascript文件：

```
<script src="file.js" />
```

- 在标记中的事件中注入代码

```
<button onclick="javascript: /*code*/">press me</button>
```

Javascript语法

与C++和Java的语法类似但是要更加简单，eg：

```
var my_number = 10;
my_string = "hello";
var my_array = [10,20,"name",true];
var my_object = { name: "javi", city: "Barcelona" };

function say( str )
{
        for(var i = 0; i < 10; ++i)
        console.log(" say: " + str );
}
```

Javascript例子

```
<html>
  <body>
    <h1>This is a title</h1>
    <script>
      var title = document.querySelector("h1");
      title.innerHTML = "This is another title";
    </script>
  </body>
</html>
```

Javascript API

JavaScript有丰富的API库去帮助做许多的工作，比如：

- 访问 DOM (HTML 节点)
- 执行 HTTP 请求
- 播放视频和声音
- 检测用户操作(鼠标移动，按键)
- 启动线程
- 进入图形处理器，获取网络摄像头图像
- ……

同时API库还在持续更新中，想要知道更多可以访问网页API库

检索元素

使用不同的方法从 DOM (HTML 树)中获取元素：

- 爬行HTML 树(从主体开始，遍历其子树)
- 使用选择器(如CSS)
- 附加事件侦听器(在执行某些操作时调用函数)

遍历DOM（文档对象模型）

通过 javascript 你可以访问不同的变量来获得网站的信息：

- document：DOM信息（HTML）

- window：浏览器窗口

document变量允许对树进行爬行 ：

```
document.body.children[0] //返回 body 标记中的第一个节点
```

使用选择器

可以使用选择器检索元素：

```
var nodes = document.querySelectorAll("p.intro");
```

Nodes为一个包含 web 中所有 < p class =“ intro”> 节点的数组

或者，如果我们已经有了一个节点，我们想要搜索内部：

```
var node = mynode.querySelectorAll("p.intro")
```

修改节点

从JS我们可以修改属性

```
mynode.id = “intro”;  //设置id
mynode.className = “important”;   //设置类别
mynode.classList.add(“good”);  //添加至当前类
```

改变内容

```
mynode.innerHTML = “<p>text to show</p>”;  //改变内容
```

修改样式（CSS）

```
mynode.style.color = “red”;   //修改任何css属性
```

添加一个节点的行为

```
mynode.addEventListener("click", function(e) {
        //do something
}
```

创建节点

创建元素：

```
var element = document.createElement("div");
```

将元素连接到DOM上：

```
document.querySelector("#main").appendChild( element );
```

将其从父节点中移除：

```
var element = document.querySelector("foo");
element.parentNode.removeChild( element );
```

可以简单地复制一个元素

```
var cloned = element.cloneNode(true);
```

网页示例

HTML in index.html

```
<link href="style.css" rel="stylesheet"/>
<h1>Welcome</h1>
<p>
     <button>Click me</button>
</p>
<script src="code.js"/>
```

CSS in style.css

```
h1 { color: #333; }
button {
     border: 2px solid #AAA;
     background-color: #555;
}
```

Javascript in code.js

```
//从 DOM 获取按钮
var button = document.querySelector("button");

//attach and event when the user clicks it
button.addEventListener("click", myfunction);

//按钮被按下时响应的函数

function myfunction()
{
     //显示一个弹出窗口
     alert("button clicked!");
}
```

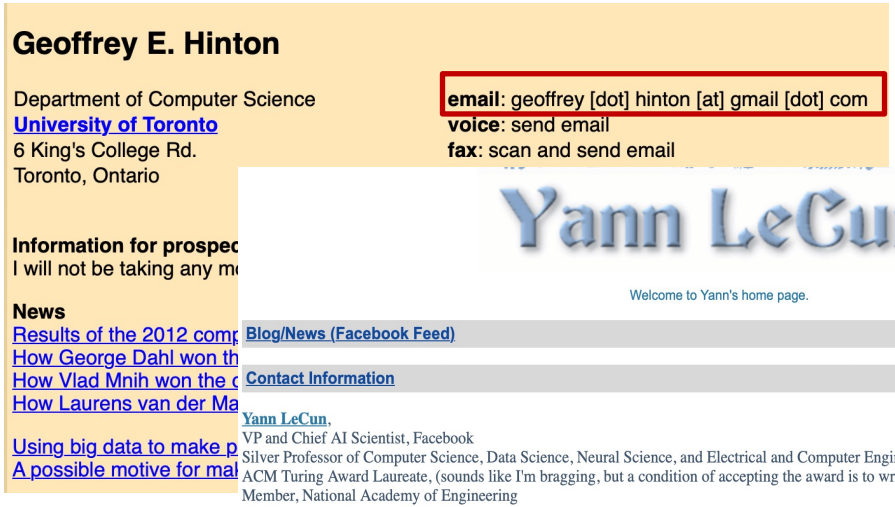
正则表达式

正则表达式

正则表达式是处理字符串的强大工具

- 一种基于**字符**（***character-based***）的语言
- 基于一定的语法规则构造模式（***patterns***）
- 有效实现文本匹配、检索、替换

正则表达式



Geoffrey E. Hinton

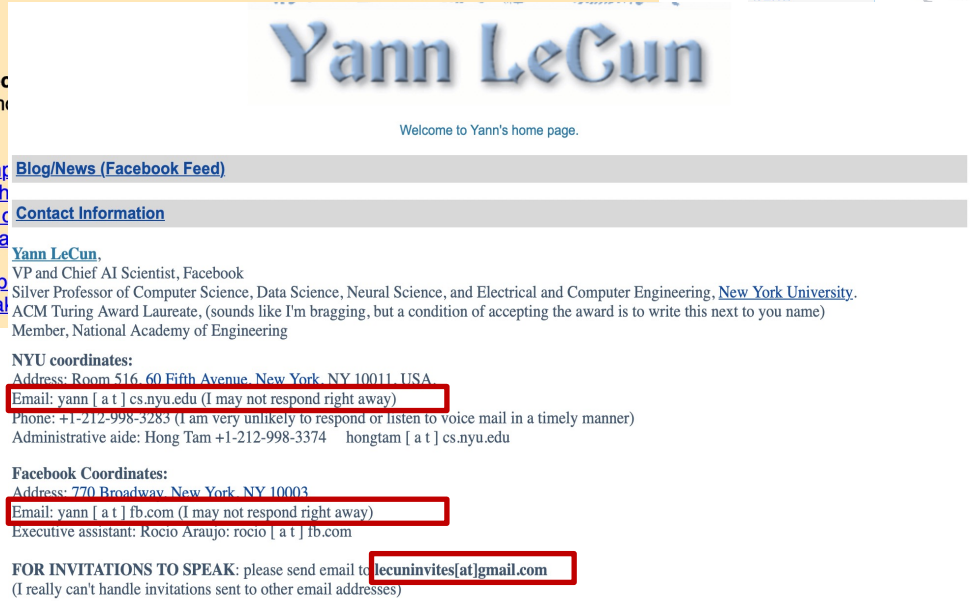
Department of Computer Science
University of Toronto
6 King's College Rd.
Toronto, Ontario

email: geoffrey [dot] hinton [at] gmail [dot] com
voice: send email
fax: scan and send email

Information for prospec
I will not be taking any m

News
Results of the 2012 comp
How George Dahl won th
How Vlad Mnih won the c
How Laurens van der Ma

Using big data to make p
A possible motive for mal



Yann LeCun

Welcome to Yann's home page.

Blog/News (Facebook Feed)

Contact Information

Yann LeCun,
VP and Chief AI Scientist, Facebook
Silver Professor of Computer Science, Data Science, Neural Science, and Electrical and Computer Engineering, New York University.
ACM Turing Award Laureate, (sounds like I'm bragging, but a condition of accepting the award is to write this next to you name)
Member, National Academy of Engineering

NYU coordinates:
Address: Room 516, 60 Fifth Avenue, New York, NY 10011, USA
Email: yann [a t] cs.nyu.edu (I may not respond right away)
Phone: +1-212-998-3283 (I am very unlikely to respond or listen to voice mail in a timely manner)
Administrative aide: Hong Tam +1-212-998-3374 hongtam [a t] cs.nyu.edu

Facebook Coordinates:
Address: 770 Broadway, New York, NY 10003
Email: yann [a t] fb.com (I may not respond right away)
Executive assistant: Rocio Araujo: rocio [a t] fb.com

FOR INVITATIONS TO SPEAK: please send email to **lecuninvites[at]gmail.com**
(I really can't handle invitations sent to other email addresses)

维基百科
自由的百科全书

杭州市 [编辑]

维基百科，自由的百科全书
（重定向自杭州）

"武林"重定向至此。关于其他用法，请见"武林 (消歧义)"。

杭州市
杭
地级市

查找匹配、解析

正则表达式是一种基于字符的语言

标记字符

- 元字符
- 量词

子表达式

- 嵌套子表达式

什么是基于字符的语言？

正则表达式像Python语言一样定义了一些具有特定语义的关键字，只不过这些关键字大多都是字符。

我们称这些字符为**标记字符（*marker character*）**

即它们出现时并不代表字符本身，而是有固定的语义。

除了元字符和量词外的其他字符在正则表达式中出现时都代表自己本身。

```
import re
```

正则表达式语法要素：标记字符

元字符

每个元字符默认匹配一位字符串

`.`	匹配除换行符以外的任意字符
`^`	匹配字符串的开始
`$`	匹配字符串的结尾
`a|b`	匹配字符a或字符b
`\w`	匹配字母或数字或下划线
`\W`	匹配非字母或数字或下划线
`\s`	匹配任意空白符
`\S`	匹配非空白符
`[aeiou]`	匹配字符组中的字符（注：可由区间构造字符组，如`[a-z0-9]`）
`[^XYZ]`	匹配除了字符组中字符的所有字符
`(`	匹配子表达式开始
`)`	匹配子表达式结束

正则表达式语法要素：标记字符

量词

量词修饰它的前一位字符；控制前面的元字符出现的次数

`*`	**重复0次或更多次**
`+`	**重复1次或更多次**
`?`	**重复0次或1次（出现或不出现）**
`{n}`	**重复n次**
`{n,}`	**重复n次或更多**
`{n,m}`	**重复n次到m次**

量词也可以修饰量词

`.*`	**贪婪匹配（*表示尽可能多的去匹配）**
`.*?`	**惰性匹配（?修饰*，表示让*取一个可以匹配上的最小量值）**

正则表达式语法要素：标记字符

量词

文本	这是第一堂课，这是第二堂课
正则表达式	匹配结果
.*课	这是第一堂课，这是第二堂课
.*?课	这是第一堂课

```
import re
text = "这是第一课堂"
```

```
pattern = re.compile(".*课")
m = pattern.search(text)
```

```
m = re.search(".*课", text)
```

```
if m is not None:
    print(m.group())
```

量词也可以修饰量词

.*　贪婪匹配（*表示尽可能多的去匹配）
.*?　惰性匹配（?修饰*，表示让*取一个可以匹配上的最小量值）

正则表达式语法要素：标记字符

转义字符

当需要匹配用于正则匹配模式的特殊字符（例如：.）时，就需要用到转义符了，即在要匹配的特殊字符前面加反斜线 \ 转义一下即可（例如：\.）。

文本	1人民币=3585.67越南盾
正则表达式	re.search匹配结果
\d.\d	358
\d\.\d	5.6

正则表达式语法要素：子表达式

嵌套子表达式

- 包裹在圆括号内的子表达式作为一个整体参与匹配
- 执行匹配后，除了总匹配结果，子表达式匹配结果也会被存下来
- 给子表达式(分组)起名字：(?P<分组名字>子正则表达式)

正则表达式语法要素：子表达式

嵌套子表达式

- 给子表达式(分组)起名字：(?P<分组名字>子正则表达式)

```
1.import re
2.s = """<div class="西游记"><span id="10010">中国联通</span></div>"""
3.# re.S这个flag的作用是：让元字符.能匹配换行符，怕断行
4.COMPILED_REGX = re.compile(r'<span id="(?P<id>\d+)">(?P<name>\w+)</span>', re.S)
5.result = COMPILED_REGX.search(s)
6.print(result.group())
7.# 获取id组的内容
8.print(result.group("id"))    # 10010
9.# 获取name组的内容
10.print(result.group("name"))    # 中国联通
```

字符串的检索、匹配和替换

- **re.search & re.findall**

  re.search全文去检索匹配，找到一个结果就返回，返回match对象。re.search可以返回匹配正则表达式的第一个内容，但是如果想要获取匹配正则表达式的所有内容就要借re.findall方法了。

- **re.match**

  从头开始匹配，找到一个结果就返回，返回match对象，好比默认在正则表达式最开始加了个^元字符。

- **re.sub**

  可以借助re.sub来进行文本内容替换

  例如把一串文本中的所有数字都去掉：content = re.sub(‘\d+’, ‘’, content)