

Lecture 2

HTML 基础

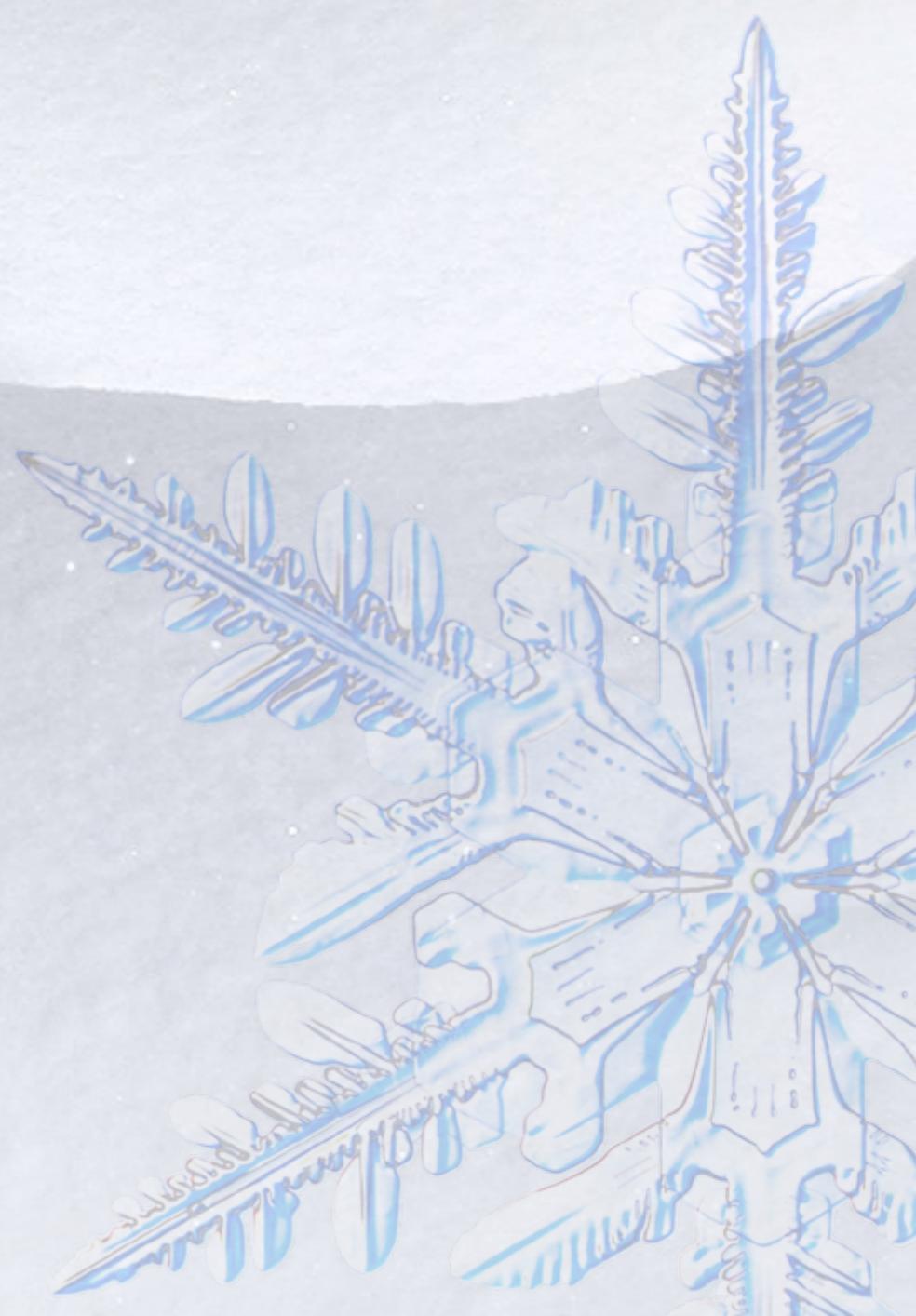


目录

❄️ **HTML概述**

❄️ **表单**

❄️ **Web标准**



HTML

- ❖ Hypertext Markup Language (HTML) , 全称为超文本标记语言，是一种标记语言。它包括一系列标签. 通过这些标签可以将网络上的文档格式统一，使分散的Internet资源连接为一个逻辑整体。
- ❖ HTML (超文本标记语言——HyperText Markup Language) 是构成 Web 世界的一砖一瓦。它定义了网页内容的含义和结构。除 HTML 以外的其它技术则通常用来描述一个网页的表现与展示效果 (如 CSS) , 或功能与行为 (如 JavaScript) -- MDN

HTML 版本

- 1989: 伯纳斯-李写了一份备忘录，提议建立一个基于互联网的超文本系统
- 1995: RFC 1866, HTML 2成为官方标准语言
- 1997: HTML 4
- 1999: HTML 4.01
- 2001-01: XHTML
- 2014 html5
- 2016 html5.1
- 2017 html5.2



HTML and XHTML

❄ HTML最初是一种应用程序标准通用标记语言(SGML)

- * SGML是一种非常灵活的标记语言
- * 需要一个相对复杂、宽松且通常自定义的解析器

❄ XHTML:

- * XML的应用程序，SGML的一个更严格的子集
- * 真正的XHTML文档允许使用标准XML工具执行自动化处理



标记语言

❖ 标记:

- * 在文档中嵌入代码
- * 代码被称为“标签”
- * 代码
 - * 描述结构文档
 - * 包括处理说明

❖ 标记语言:

- * 描述标签语法的计算机语言
- * 可以与其他工具一起使用来指定渲染

逻辑标记

❄ 逻辑标记:

- * 描述文档的各个部分
- * 不指定如何渲染

❄ 例如:

- * This is very important
- * This is very important

逻辑

- ❄ 呈现是客户的“决定”
- ❄ 当客户端无法展示时，会出现优雅降级
 - *

Markdown

- ❖ Markdown 是一种轻量级标记语言，它允许人们使用易读易写的纯文本格式编写文档。
 - ❖ Markdown 语言在 2004 由约翰·格鲁伯（英语：John Gruber）创建。
 - ❖ Markdown 编写的文档可以导出 HTML、Word、图像、PDF、Epub 等多种格式的文档。
 - ❖ Markdown 编写的文档后缀为 .md, .markdown。
- ## ❖ Markdown 应用
- * Markdown 能被使用来撰写电子书，如：Gitbook。
 - * 当前许多网站都广泛使用 Markdown 来撰写帮助文档或是用于论坛上发表消息。例如：GitHub、简书、reddit、Diaspora、Stack Exchange、OpenStreetMap、SourceForge等。

```
# h1 标题  
## h2 标题  
### h3 标题
```

```
## 文本样式
```

```
**This is bold text**
```

```
This is bold text
```

```
*This is italic text*
```

```
This is italic text
```

```
~~Strikethrough~~
```

```
## 代码
```

```
Inline `code`
```

```
Indented code
```

```
// Some comments  
line 1 of code  
line 2 of code  
line 3 of code
```

```
Block code "fences"
```

```
---
```

```
Sample text here...
```

```
---
```

```
Syntax highlighting
```

```
```js  
var foo = function (bar) {
 return bar++;
};
```

```
console.log(foo(5));
```

```
```
```

h1 标题

h2 标题

h3 标题
文本样式

This is bold text

This is bold text

This is italic text

This is italic text

~~Strikethrough~~

代码

Inline code

Indented code

```
// Some comments  
line 1 of code  
line 2 of code  
line 3 of code  
Block code "fences"
```

Sample text here...

Syntax highlighting

```
var foo = function (bar) {  
    return bar++;  
};
```

```
console.log(foo(5));
```

如何编写(x)HTML

❄ 编辑器

❄ 生成器

- * [https://github.com/Microsoft/ailab/tree/master/
Sketch2Code](https://github.com/Microsoft/ailab/tree/master/Sketch2Code)
- * <https://sketch2code.azurewebsites.net/>

YOUR SKETCH

Sign Up Login

Name

Last Name

Phone

E-mail

Password

Confirm Password

I agree to Terms and Conditions

YOUR HTML

Sign Up Login

Name

Last Name

Phone

E-mail

Password

Confirm Password

I agree to Terms and Conditions

YOUR SKETCH

Sign Up Login

Name

Last Name

Phone

E-mail

Password

Confirm Password

I agree to Terms and Conditions

YOUR HTML

Sign Up Login

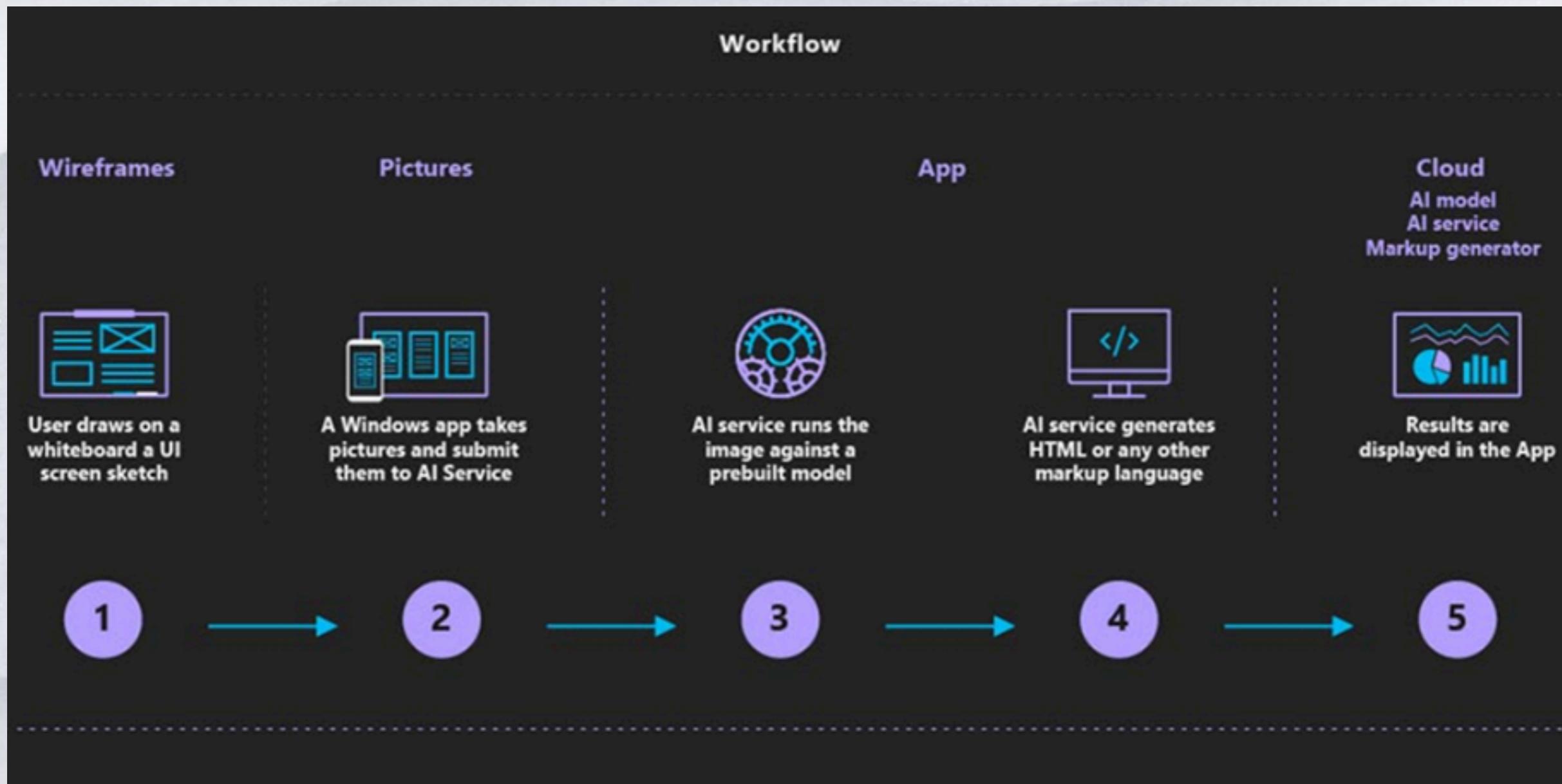
Name

Phone

Continue Password

I agree to Terms and Condition

Sketch2Code处理流程



↑ Upload & results

Azure Cloud Platform

⊗ Azure Cloud AI Webservices

Sketch2Code Web
(MVC Application)



HTML Generation

Async Parallel Web Calls

Azure Functions



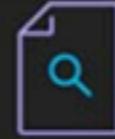
AI Orchestrator

Azure Blob Storage



Payload & image storage

Algorithm



Layout Identification

Customvision.AI



Object Detection

Computer Vision API



Handwritten Model

Sketch2Code Architecture Diagram

HTML 基础

*XHTML 或 HTML 文档组成

- * DOCTYPE

- * 使用的文档类型定义DTD

- * Head

- * 元信息
 - * 只有<title> 是必须的

- * Body

- * 需要呈现的内容

XHTML网页的结构

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN" http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
    <head>
        <title>Hello World</title>
    </head>
    <body>
        page contents
    </body>
</html>
```

```
<!DOCTYPE html>
<html>
    <body>
        <h1>My First Heading</h1>
        <p>My first paragraph.</p>
    </body>
</html>
```

HTML 元素详解

- * 开始标签（Opening tag）：包含元素的名称（本例为 p），被大于号、小于号所包围。表示元素从这里开始或者开始起作用 —— 在本例中即段落由此开始。
- * 结束标签（Closing tag）：与开始标签相似，只是其在元素名之前包含了一个斜杠。这表示着元素的结尾 —— 在本例中即段落在此结束。初学者常常会忘记包含结束标签的错误，这可能会产生一些奇怪的结果。
- * 内容（Content）：元素的内容，本例中就是所输入的文本本身。
- * 元素（Element）：开始标签、结束标签与内容相结合，便是一个完整的元素。



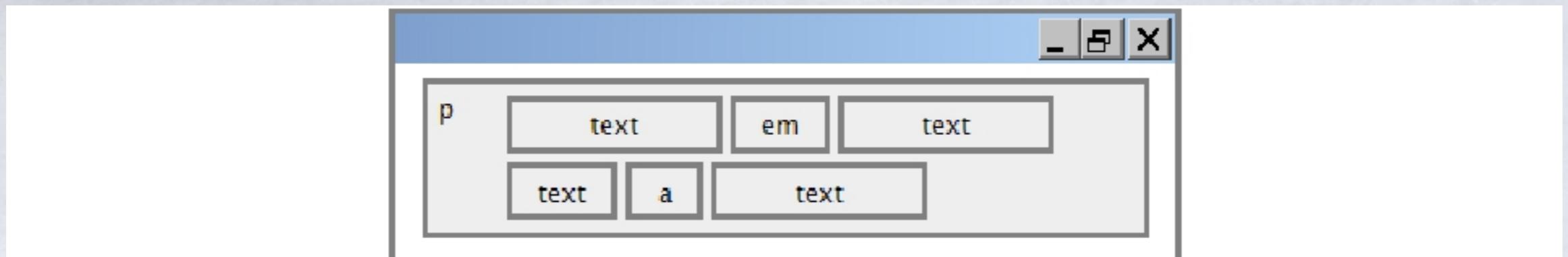
块级元素或行内元素

✿ 块级元素占据其父元素（容器）的整个水平空间，垂直空间等于其内容高度，因此创建了一个“块”。

- * 例如: paragraphs, lists, table cells
- * 通常浏览器会在块级元素前后另起一个新行

✿ 行内元素只占据它对应标签的边框所包含的空间。

- * 例如: bold text, code fragments, images
- * 浏览器允许许多行内元素出现在同一行上
- * 必须嵌套在块级元素中



注释: <!-- ... -->

❄️ 注释或“注释掉”文本

```
<!-- My web page, by Student SS 330, Spring 2048 -->  
<p>SS courses are <!-- NOT --> a lot of fun!</p>
```

HTML

SS courses are a lot of fun!

output

❄️ 不能嵌套

❄️ 注释的使用

- * 许多网页根本没有注释

网页标题: <title>

❄ 当前网页的标题

```
<title>Chapter 2: HTML Basics</title>
```

❄ 用于:

- * 定义浏览器工具栏中的标题
- * 提供页面被添加到收藏夹时的标题
- * 显示在搜索引擎结果中的页面标题

❄ 标题长度受限

❄ SEO权重大

<meta>

```
<meta name="description" content="introduction of XXX" />
```

```
<meta http-equiv="Content-Type" content="text/html;  
charset=gbk" />
```

- ❄ 用于提供关于HTML文档的元数据
- ❄ <meta>标签只能出现在<head>里
- ❄ <meta>标签通常用于给出网页描述、关键词、文档作者、最后修改日期等信息。

<meta>实例

```
<meta http-equiv="Content-Type" content="text/html;  
charset=iso-8859-1" />  
<meta name="description" content="Authors' web site for  
Building Java Programs." />  
<meta name="keywords" content="java, textbook" />
```

* name

- * author
- * description
- * keywords
- * generator
- * revised

* http-equiv

- * content-type
- * expires
- * refresh

字符编码

```
<meta charset="utf-8">
```

❄ 字符集

- * ASCII(basic 7b, extension 1B),
- * iso-8859-1/latin-1 (West Europe, 1B)
- * GB2312 (2B, Simplified Chinese)
- * GBK(2B, S. & T. Chinese)
- * GB18030 (1,2,4B, Eastern Asia)
- * Unicode (650 languages)
 - * UTF-8 (1,2,3,4B , Chinese 3B)
 - * UTF-16 (2B, 4B, Chinese 2B)
 - * UTF-32 (4B, future)
- * UCS
 - * UCS-2 (2B, comparable with UTF-16)
 - * UCS-4 (4B, future)

<h1> - <h6> 标签

- ❄ <h1> - <h6> 标签被用来定义 HTML 标题。
- ❄ 用标题来呈现文档结构是很重要的
 - * <h1> 定义重要等级最高的标题。<h6> 定义重要等级最低的标题。



```
<h1>这是一个标题。</h1>
<h2>这是一个标题。</h2>
<h3>这是一个标题。</h3>
```

<p>



段落

```
<p>You're not your job. You're not how much money you have in the bank.  
You're not the car you drive. You're not the contents of your wallet. You're  
not your khakis. You're the all-singing, all-dancing crap of the world.</p>
```

html

```
You're not your job. You're not how much money you have in the  
bank. You're not the car you drive. You're not the contents of your  
wallet. You're not your khakis. You're the all-singing, all-dancing crap  
of the world.
```

output

✿ 插入一个简单的换行符

```
<p>Teddy said it was a hat, <br /> So I put it on.</p> <p>Now Daddy's  
sayin', <br /> Where the heck's the toilet plunger gone?</p>
```

Teddy said it was a hat,
So I put it on.

Now Daddy's sayin',
Where the heck's the toilet plunger gone?

✿
 标签是一个空标签，意味着它没有结束标签。

<a>

❄ 定义超链接

```
<p>
  Search
  <a href="http://www.google.com/">Google</a> or our
  <a href="lectures.html">Lecture Notes</a>
</p>
```

Search Google or our Lecture Notes.

- ❄ <a> 元素最重要的属性是 href 属性，它指定链接的目标。
- ❄ 请使用 CSS 来改变链接的样式。

<a>



Click here to check your course schedule

Please check your course schedule

**Course Schedule (please check yours before
March 15!)"**



图像

- ❄ 是空标签，只包含属性，并且没有闭合标签

```

```

- ❄ src 指 "source"。源属性的值是图像的 URL 地址。
- ❄ alt 属性用来为图像定义一串预备的可替换的文本。

此外

- ✿ 通过在 `<a>` 标签中嵌套 `` 标签，给图像添加到另一个文档的链接。

```
<a href="http://theonering.net/">  
      
</a>
```

图像格式

❄️ GIF

- * 透明、无损、256
- * 用于动画

❄️ JPEG

- * 使用范围最广
- * 有损
- * 背景图、轮播图或者商品的banner图

❄️ PNG

- * 真彩色和调色板、透明、无损
- * 文件体积大
- * 颜芻数少的单帧图像，图标，LOGO

❄️ webp

- * 有损、压缩率高，同时支持透明度和动画
- * 浏览器兼容性

❄️ SVG

- * LOGO

,

- ❄ em: 呈现为被强调的文本
- ❄ strong: 定义重要的文本

```
<p>  
    HTML is <em>really</em>,  
    <strong>REALLY</strong> fun!  
</p>
```

HTML is *really*, **REALLY** fun!

- ❄ em vs. i, strong vs. b

- * <i>无强调或着重意味的斜体 (italic) , 比如生物学名、术语、外来语 (比如「*de facto*」这样的英语里常用的拉丁语短语)
- * 无强调或着重意味的粗体 (bold) , 比如文章摘要中的关键词、评测文章中的产品名称、文章的导言.....

嵌套的标签



```
<p>  
    HTML is <em>really,  
    <strong>REALLY</em> lots of </strong> fun!  
</p>
```

- ❄ 标签必须正确嵌套
 - * 结束标记必须与最近打开的标记匹配
- ❄ 浏览器可能会正确地呈现它，但它是无效的XHTML

Table: <table>, <tr>, <td>, <th>, <caption>

```
<table>  
<caption>Smart Guys</caption>  
<tr><th>name</th><th>gender</th></tr> <tr><td>Bill</td><td>male</td></tr> <tr><td>Susan</td><td>female</td></tr>  
</table>
```

Smart Guys

name	gender
Bill	male
Susan	female

* 不要用于布局!

- * 结构混乱，不清晰
- * CSS更强大，访问性更好，能在范围更广的设备上运作，如手机
- * 机器难以理解，不利于SEO
- * 基于CSS的布局页面比表单布局更小更简单

<blockquote>



块元素

```
<p>As Lincoln said in his famous Gettysburg Address:</p>
<blockquote>
<p>Fourscore and seven years ago, our fathers brought forth on this continent a new
nation, conceived in liberty, and dedicated to the proposition that all men are created
equal.</p>
</blockquote>
```

As Lincoln said in his famous Gettysburg Address:

*Fourscore and seven years ago, our fathers brought forth
on this continent a new nation, conceived in liberty, and
dedicated to the proposition that all men are created equal.*

output

行内引用<q>

```
<p>Quoth the Raven, <q>Nevermore.</q></p>
```

Quoth the Raven, “Nevermore”.

※ 为何不像下面的方式?

* <p>Quoth the Raven, "Nevermore."</p>

不使用“ 的原因是:

- * XHTML不应该直接包含引号字符，应该写成转义"
- * 使用 <q>的话可以使用CSS样式

HTML 字符实体

- ❄ HTML 中的预留字符必须被替换为字符实体
- ❄ 一些在键盘上找不到的字符也可以使用字符实体来替换

character (s)	entity
<>	< >
é è ñ	´ è ñ
TM ©	™ ©
π δ Δ	π δ Δ
И	И
" &	" &

- ❄ 如何在网页中显示文本 & ?

HTML的显示

- * 要在Web页面中显示链接文本，必须对其特殊字符进行如下所示的编码

```
&lt;p&gt; &lt;a href="http://google.com/search?  
q=marty&ie=utf-8&aq=t"&gt; Search  
Google for Marty &lt;/a&gt; &lt;/p&gt;
```

HTML

```
<p> <a href="http://google.com/search?q=marty&ie=utf-8&aq=t"> Search  
Google for Marty </a> </p>
```

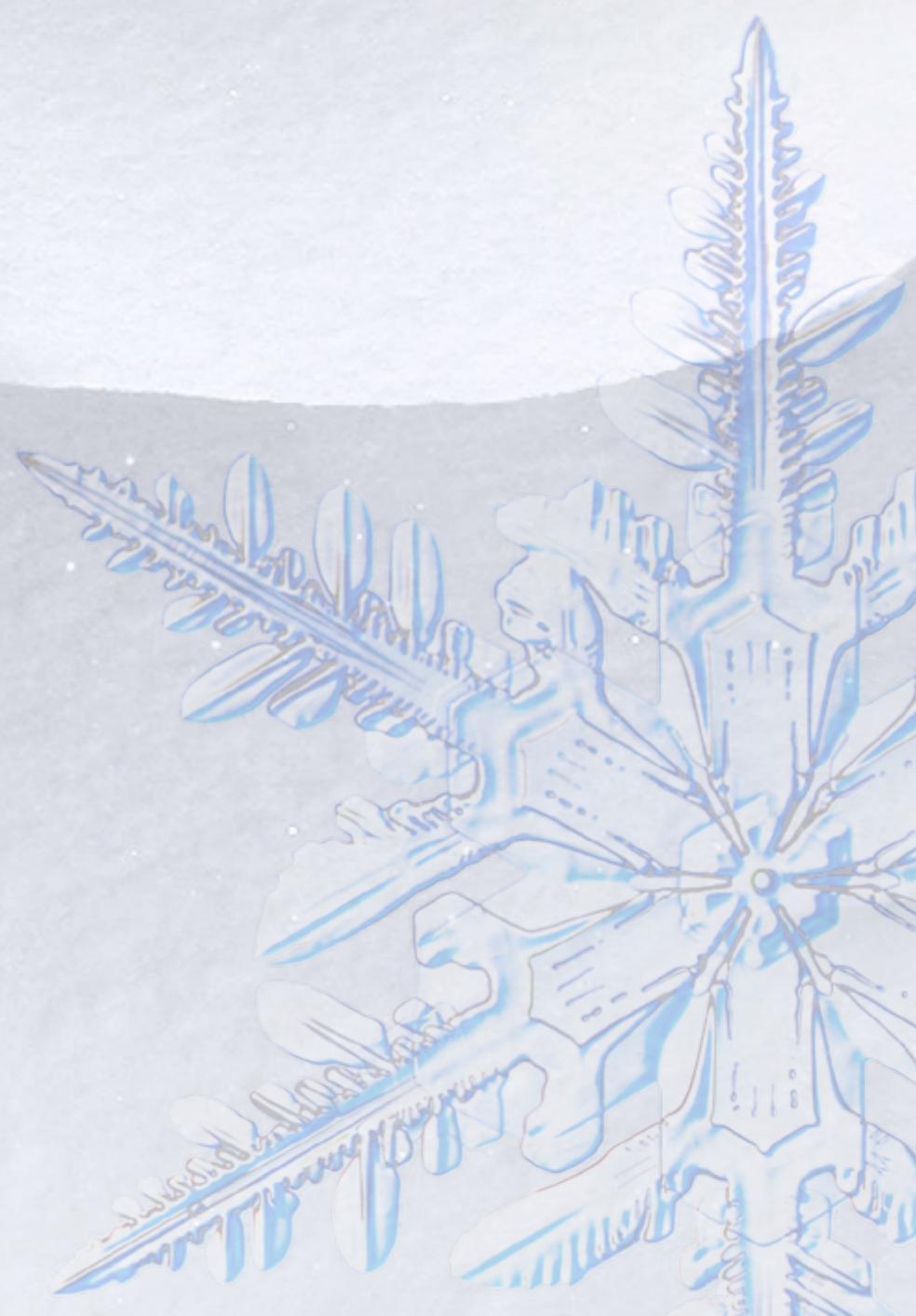
output

目录

❄ HTML概述

❄ 表单

❄ Web标准



表单

- ✿ <form>: HTML 表单用于收集用户的输入信息
- ✿ HTML 表单表示文档中的一个区域，此区域包含交互控件，将用户收集到的信息发送到 Web 服务器。
 - * 基本语法: <form parameters> ...form elements... </form>
 - * 表单元素允许用户在表单中输入内容，比如：文本域 (textarea)、下拉列表 (select)、单选框 (radio-buttons)、复选框 (checkbox) 等等
 - * 包含提交按钮
 - * 当用户单击确认按钮时，表单的内容会被传送到服务器。表单的动作属性 action 定义了服务端的文件名。

表单例子

- 必须将表单的控件包装在块元素中，比如div, fieldset等

```
<form>
  <div class="form-group">
    <label for="inputEmail">Email</label>
    <input type="email" class="form-control" id="inputEmail" placeholder="Email">
  </div>
  <div class="form-group">
    <label for="inputPassword">Password</label>
    <input type="password" class="form-control" id="inputPwd" placeholder="Password">
  </div>
  <div class="checkbox">
    <label><input type="checkbox"> Remember me</label>
  </div>
  <button type="submit" class="btn">Login</button>
</form>
```

Email Email

Password Password

Remember me

Login

<form> 标签

❄ 表单属性说明如何处理用户输入

- * **action="url"** (必须)
 - * 指定单击Submit按钮时将数据发送到哪里
- * **method="get"** (缺省)
 - * 将表单数据以名称/值对的形式附加到 URL 中
 - * URL 的长度是有限的 (大约 3000 字符)
 - * 绝不要使用 GET 来发送敏感数据! (在 URL 中是可见的)
 - * 对于用户希望加入书签的表单提交很有用
 - * GET 更适用于非安全数据, 比如在 Google 中查询字符串
- * **method="post"**
 - * 将表单数据附加到 HTTP 请求的 body 内 (数据不显示在 URL 中)
 - * 没有长度限制
 - * 通过 POST 提交的表单不能加入书签

<input>

- ❄ <input> 标签规定了用户可以在其中输入数据的输入字段。
- ❄ <input> 元素在 <form> 元素中使用，用来声明允许用户输入数据的 input 控件。
- ❄ 输入字段可通过多种方式改变，取决于 type 属性。

```
<form action="demo_form.php">
  First name: <input type="text" name="fname"><br>
  Last name: <input type="text" name="lname"><br>
  <input type="submit" value="提交">
</form>
```

例子

- ✿ <input type="text" name="firstname" />
- ✿ <input type="radio" name="sex" value="male" />
Male
- ✿ <input type="checkbox" name="bike" />
- ✿ <input type="password" name="pwd" />

关于<input>的思考

❄ 输入的类型这么多，为什么不使用元素呢？

- * <input type="text" ... /> → <text/> or <text></text>
- * <input type="checkbox" ... /> → <checkbox ... />

❄ 事实上，当表单在1996年首次被设计并引入到html中时，
这只是一个糟糕的设计决策，我们一直遵循它到目前为止...

❄ 另一个缺陷

- * checked="checked" ..., 怪异？

❄ 所以：

- * 现实从来都不完美
- * 但是我们会尽最大努力使它变得完美

隐藏字段

* <input type="hidden" name="hiddenField" value="xxx"><-- right there, don't you see it?

A hidden field: <-- right there, don't you see it?

* 作用

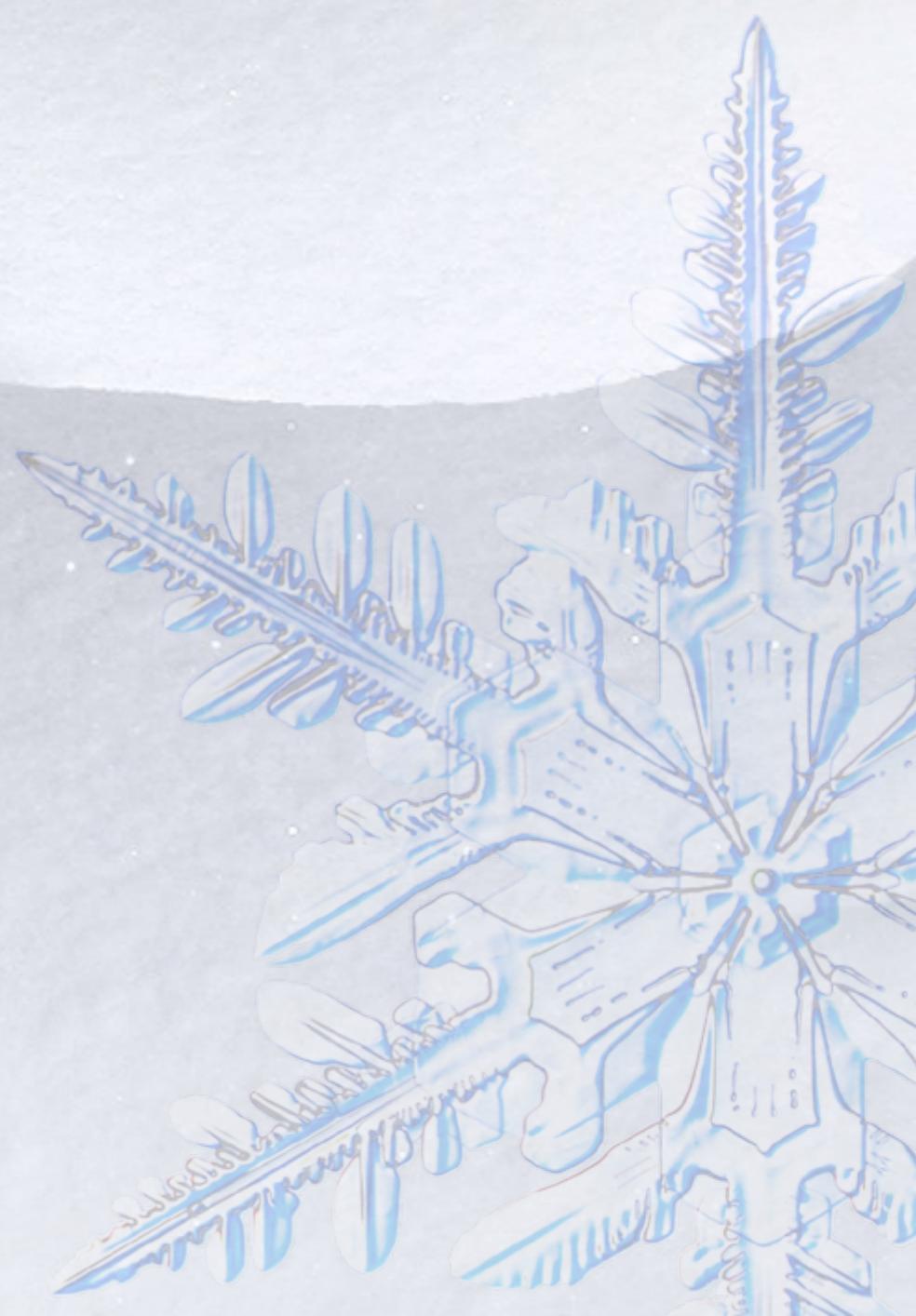
- * 隐藏域在页面中对于用户是不可见的，在表单中插入隐藏域的目的在于收集或发送信息，以利于被处理表单的程序所使用。浏览者单击发送按钮发送表单的时候，隐藏域的信息也被一起发送到服务器。
- * 在提交表单时提交上来以确定用户身份，如sessionkey等等
- * form里有多个提交按钮，分清是那一个按钮提交的
- * 一个网页中有多个form，是不能同时提交的，但有时这些form确实相互作用，就可以在form中添加隐藏域来使它们联系起来

目录

❄ HTML概述

❄ 表单

❄ Web标准



Web标准

✿ 为什么使用Web标准?

- * 更严格和结构化的语言
- * 不同浏览器之间的兼容性更强
- * 更有可能让页面在以后也能正确显示
- * 可以与SVG(图形)、MathML、MusicML等其他XML数据进行交换。

XHTML 1.0 vs HTML 4.01

❄ 文档结构

- * XHTML DOCTYPE 是强制性的
- * <html> 中的 XML namespace 属性是强制性的
- * <html>、<head>、<title> 以及 <body> 也是强制性的

❄ 元素语法

- * XHTML 元素必须正确嵌套
- * XHTML 元素必须始终关闭
- * XHTML 元素必须小写
- * XHTML 文档必须有一个根元素

❄ 属性语法

- * XHTML 属性必须使用小写
- * XHTML 属性值必须用引号包围
- * XHTML 属性最小化也是禁止的

Ref.

- * <https://www.w3schools.com/>
- * <https://www.khanacademy.org/computing/computer-programming/html-css/intro-to-html/pt/html-basics>
- * <https://classroom.udacity.com/nanodegrees/nd001-cn-preview/part/63e29129-49ba-41cf-87d6-f98246da3f24/modules/166575d1-e43b-4cbf-90da-26c36c6b3c96/lessons/cf47a211-fbc7-41fa-a9c5-3c0c6613eb05/concepts/74229205980923>

Thanks!!!

