



Woodland
Academy

4.1 HTML 基礎

- ウェブ開発
- HTML
- HTML の基本文法
- 代表的なタグ



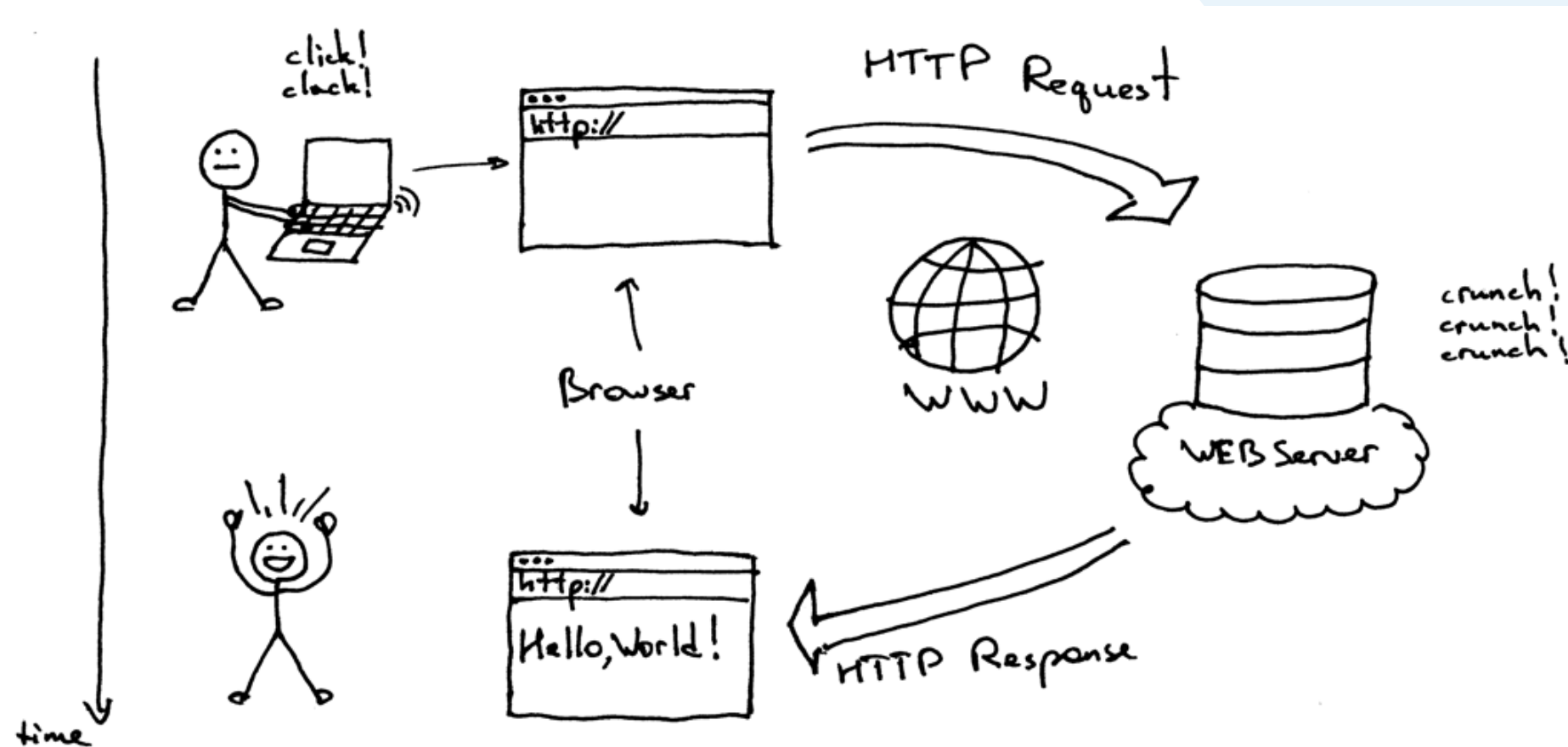
Shape Your Future

目次

- 1 ウェブ開発
- 2 HTML
- 3 HTML の基本文法
- 4 代表的なタグ

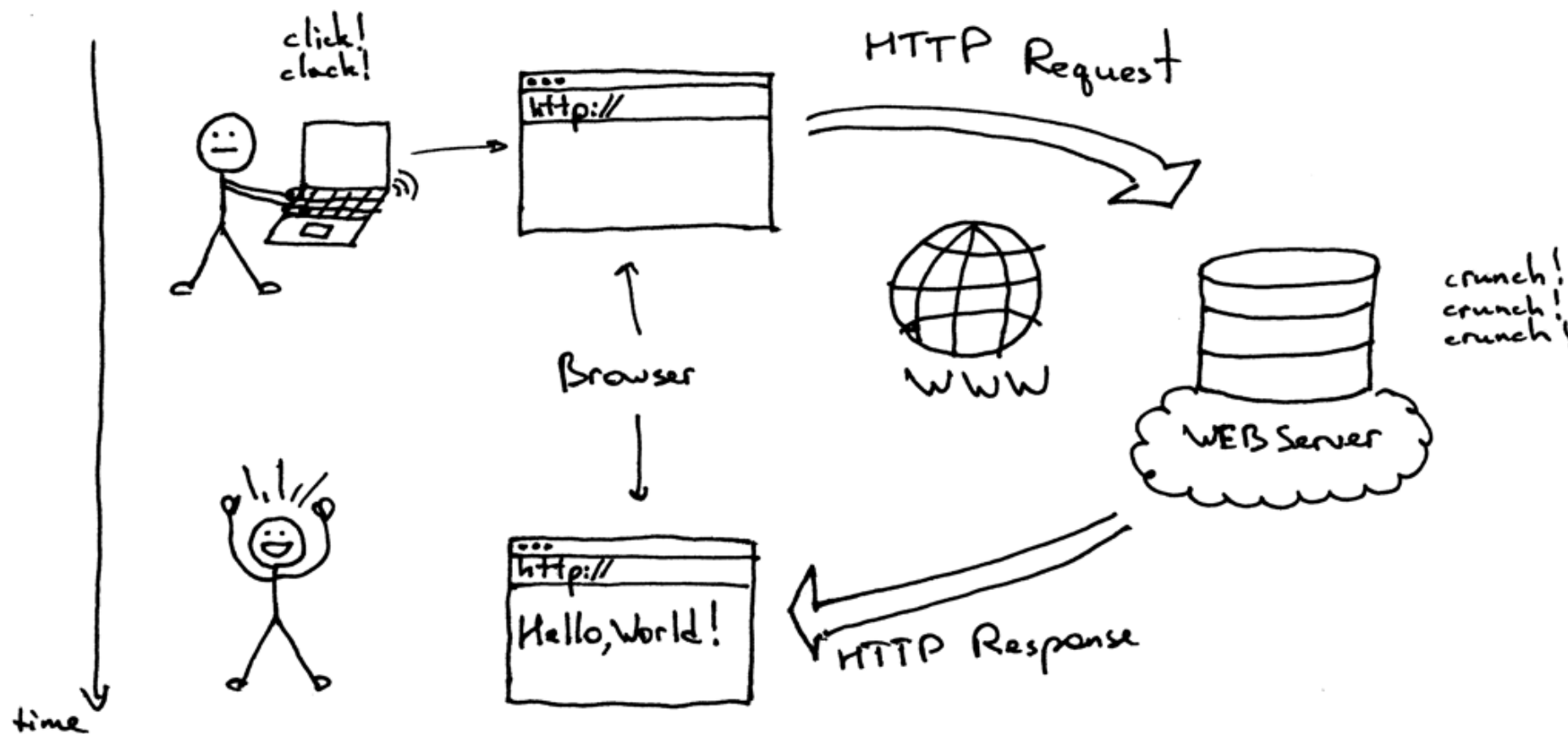
ウェブサービス

- **ウェブサービス**^[Web Service]は **HTTP プロトコル**を介してマルチメディア^[Multimedia]情報データをユーザー（**クライアント**^[Client]）に提供するサービスです。



ウェブページ

- ウェブサービスが提供するデータの基本単位は、**ウェブページ**^[Web Page]であり、各ウェブページにはテキスト、画像、音声、動画など様々な情報が含まれています。また、ウェブページに**ハイパーリンク**^[Hyperlink]を追加して、別のページに直接アクセスできるようにすることもできます。



ウェブサーバ

- **ウェブサーバ**^[Web Server]とは、ウェブページを提供する役割を担うコンピューターで、その上にウェブサーバソフトウェアが搭載されます。以下は、Netcraft による 2021 年までの主なウェブサーバソフトウェア：

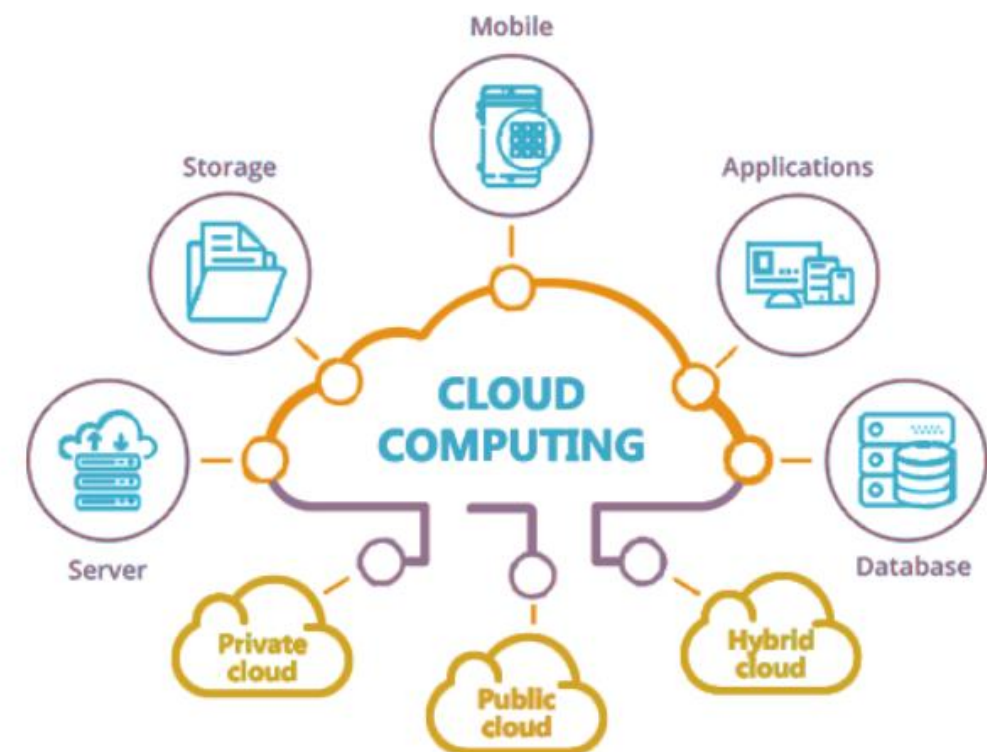
サーバ	開発した会社	市場シェア率
nginx	Nginx, Inc.	34.95%
Apache	Apache Software Foundation	24.63%
OpenResty	OpenResty Software Foundation	6.45%
Cloudflare	Cloudflare, Inc.	4.87%
IIS	Microsoft	4.00%
GWS	Google	4.00%

クラウドコンピューティング

- **クラウドコンピューティング** [Cloud Computing] とは伝統的なクライアント・サーバー方式とは異なる、新しい形態のインターネットサービスの形式です。
- クラウドサービスは、共有されたハードウェアとソフトウェアのリソースを通してユーザーにサービスを提供し、サーバー側のホルダーは、基盤となるサービスの詳細を知る必要がなくなりました。
- 代表的なクラウドサーバサービスとして、Amazon Web Services (AWS)、Google Compute Platform、Microsoft Azure などが挙げられます。

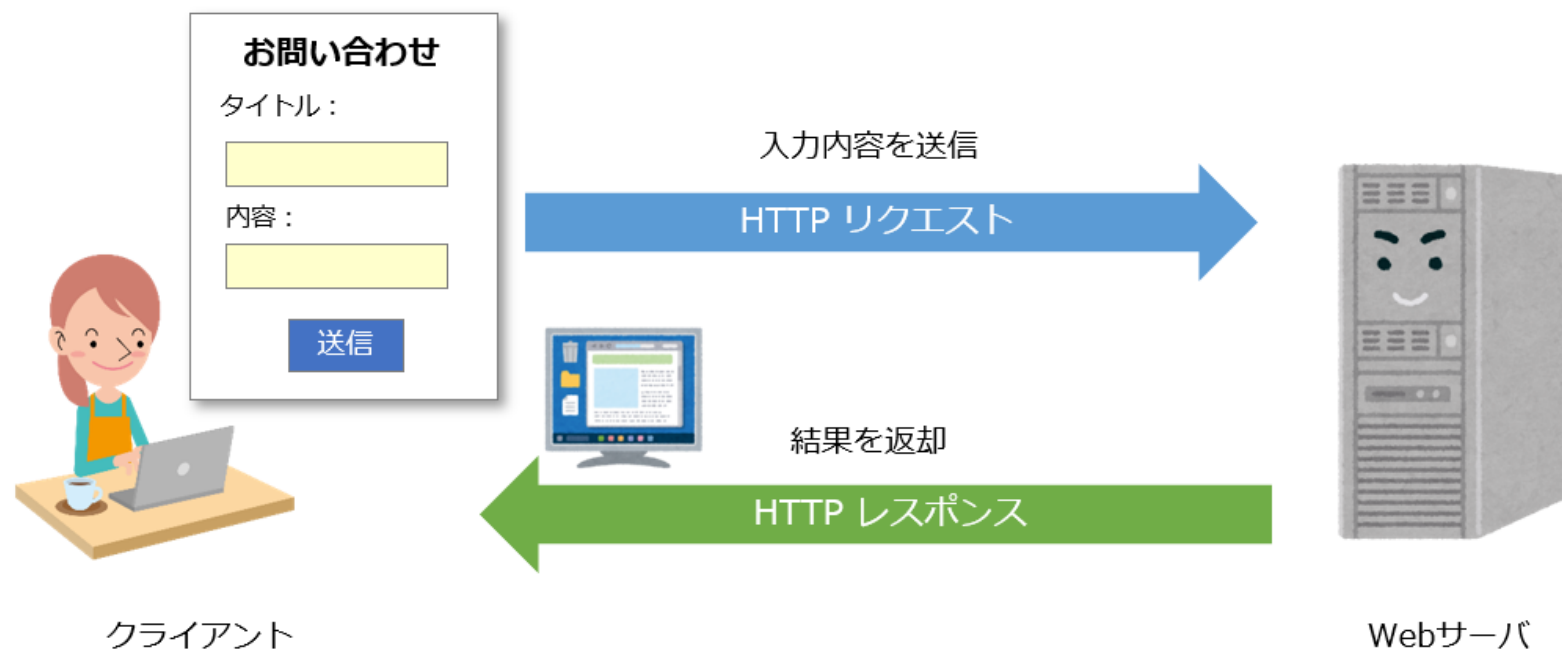
クラウドのメリット

- クラウドサービスに通してサーバを作るのは、以下のような利点があります：
 - サーバーサイドのデプロイメントにかかる学習コストを低減。
 - サーバーサイドの調整がより高速。
 - 柔軟にリソースの使用量を調整可能。
 - グローバル展開が容易になる。



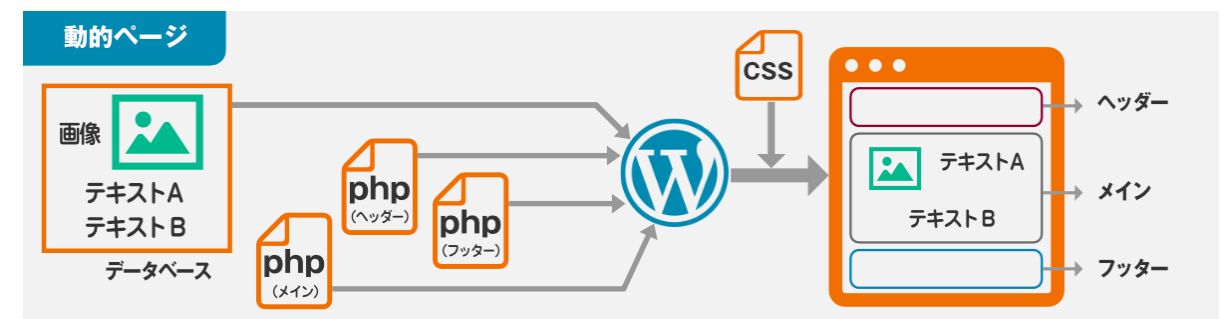
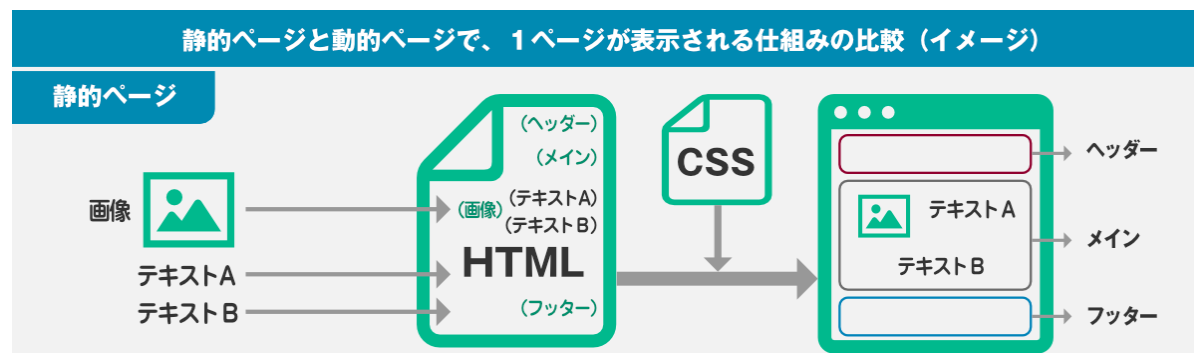
HTTP とは

- **HTTP** (HyperText Transfer Protocol) とは、クライアントとサーバ間に情報を伝送するためのプロトコル[Protocol] (規格) です。
- クライアントがウェブページを欲しいとき、ブラウザやクローラーなどのプログラムを介してサーバに **HTTP リクエスト**を送信します。サーバはリクエストを受け付け、**HTTP レスポンス**という形でページのデータを返します。



静的ページと動的ページ

- **静的ページ**^[Static Web Page] : ユーザーがサーバーに同じリクエストを送ると、クライアントは常に同じページを受け取ります。いつ誰が受け取っても、ページの内容は変わりません。
- **動的ページ**^[Dynamic Web Page] : サーバーは、クライアントからのさまざまな要求に応じて、動的にウェブコンテンツを生成します。ユーザー情報や、リクエストにある付加情報に応じて、毎回異なるウェブページを生成することができます。



参考 : <https://office7fuku.com/seiteki-douteki-koushin/>

動的ページの例

- 商品をカートに入れるシーン：

カート

1
カート
2
ログイン
3
ご注文情報入力
4
ご注文情報確認
5
ご注文完了

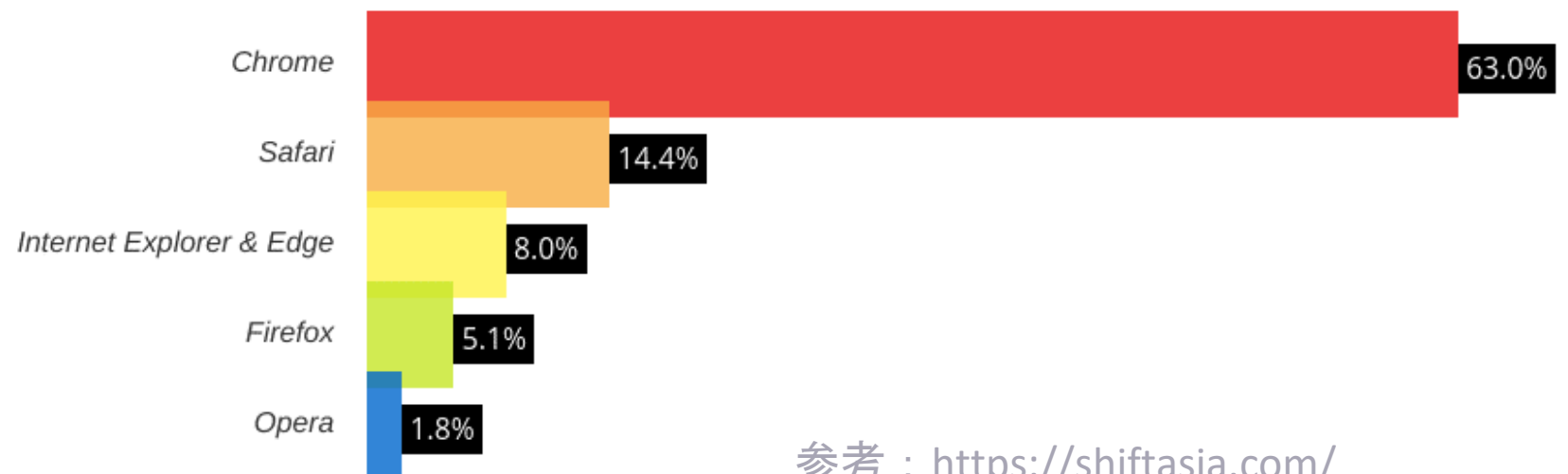
1 商品がカートに追加されました。

商品名	価格 (税込)	数量
<div> 商品コード：SMOOTHIE スムージー 削除 </div>	1,200円 (税込)	<input type="text" value="1"/>

商品点数 1点
 小計 (税込) 1,200円
 ご注文手続きへ進む
 買い物続ける
 クイック購入

ウェブブラウザ

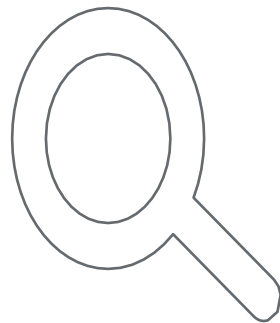
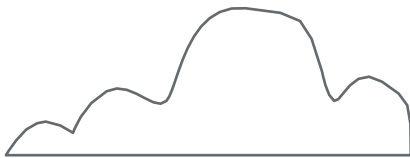
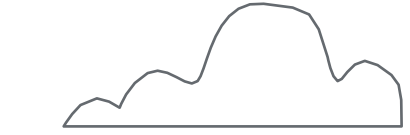
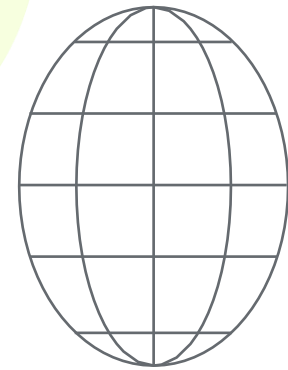
- **ウェブブラウザ**^[Browser]とはウェブページを読み取り、正しく表示することができるソフトウェアです。
- ユーザーが閲覧したいウェブページを取得するために、ウェブブラウザは、サーバーに HTTP リクエストを行います。
- 主なウェブブラウザとして、Google Chrome、Microsoft Edge、Mozilla Firefox、Safari などがあります：



参考：<https://shiftasia.com/>



Q&A



目次

- 1 ウェブ開発
- 2 **HTML**
- 3 HTML の基本文法
- 4 代表的なタグ

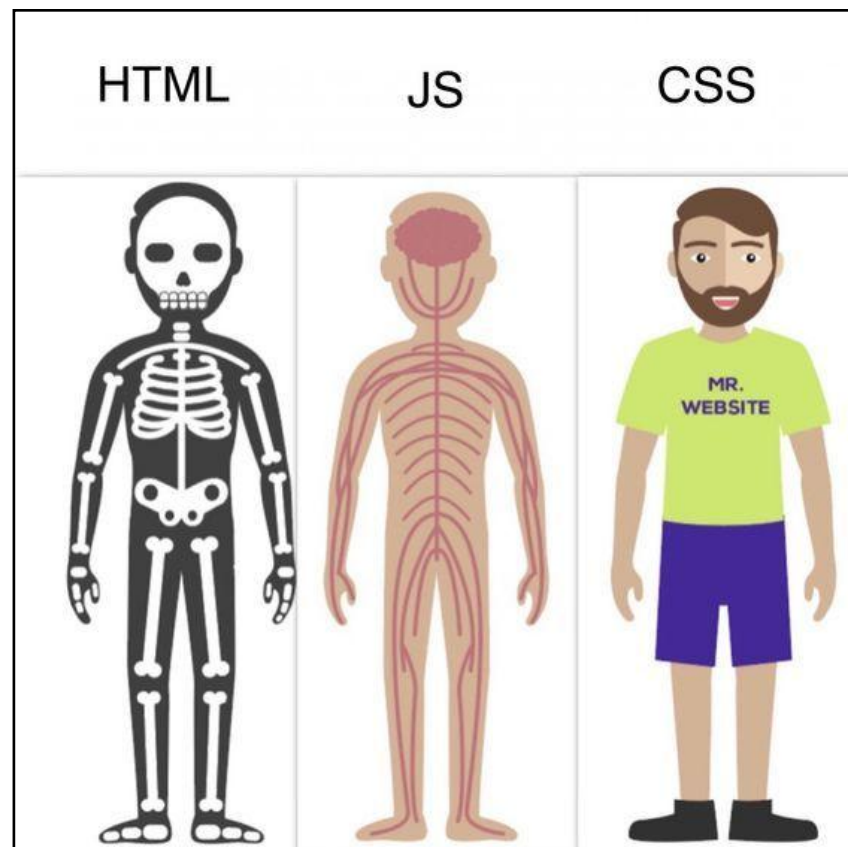
静的ページ構造

- 一般的なウェブページを作るために、**HTML**、**CSS** と **JavaScript** の 3 つの言語が使われます。
- HTML とは、ウェブページの**内容**を表現するためのマークアップ [Markup] 言語です。
- CSS とは、HTML 文書の**スタイル**を記述するスタイルシート [Style Sheet] 言語のことで、HTML 内の要素をどのように表示するかを具体的に記述したものです。
- JavaScript は、ユーザーの操作に反応して、ページの**動き方**を記述するためのプログラミング言語です。

HTML、CSS、JavaScript

- 例えば：

- HTML：テキストや画像などのコンテンツを設定。
- CSS：文字のサイズ、色、フォントなどを設定。
- JavaScript：ボタンをクリックときの反応などを設定。



HTML

- **HTML** (HyperText Markup Language) は、ページの基本構造と内容を記述するタグ言語です。
- HTML 文書は一連の**要素**から構成され、各要素はウェブページに表示されるコンテンツの一部を表現しています。
- ブラウザが HTML ファイルを開くと、各要素を読み取り、最終的に私たちが見ることのできる内容にレンダリング
[Render] します。

HTML の例

```
1 <!DOCTYPE html>
2 <html>
3   <head>
4     <title>Hello</title>
5   </head>
6
7   <body>
8     <h1>This Is A Heading</h1>
9     <p>This is a paragraph.</p>
10    <button>This is a button</button>
11  </body>
12 </html>
```

This Is A Heading

This is a paragraph.

This is a button

HTML の歴史

Year	Version
1989	Tim Berners-Lee invented www
1991	Tim Berners-Lee invented HTML
1993	Dave Raggett drafted HTML+
1995	HTML Working Group defined HTML 2.0
1997	W3C Recommendation: HTML 3.2
1999	W3C Recommendation: HTML 4.01
2000	W3C Recommendation: XHTML 1.0
2008	WHATWG HTML5 First Public Draft
2012	<u>WHATWG HTML5 Living Standard</u>
2014	<u>W3C Recommendation: HTML5</u>
2016	W3C Candidate Recommendation: HTML 5.1
2017	<u>W3C Recommendation: HTML5.1 2nd Edition</u>
2017	<u>W3C Recommendation: HTML5.2</u>

- ここでは最新バージョンの HTML Living Standard を紹介します。

HTML の作成

- HTML 文書は、Java コードと同様、通常のテキストエディタで書くことができます。テキストエディタを開き、次のコードを入力してください（インデントに注意）：

```

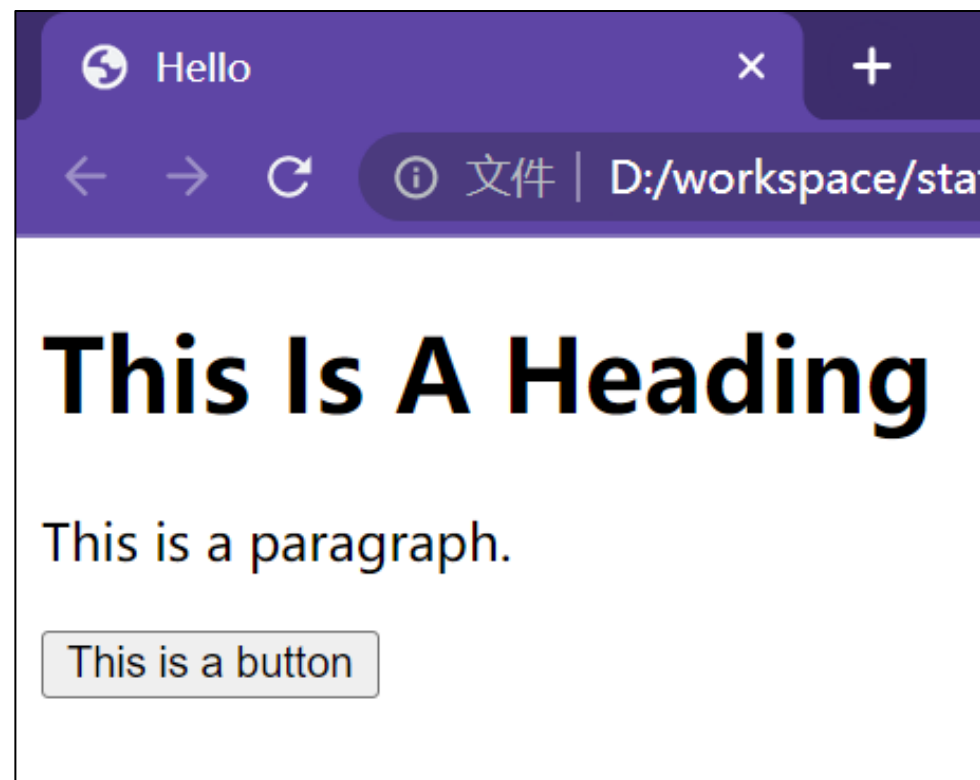
1 <!DOCTYPE html>
2 <html>
3 <head>
4   <title>Hello</title>
5 </head>
6
7 <body>
8   <h1>This Is A Heading</h1>
9   <p>This is a paragraph.</p>
10  <button>This is a button</button>
11 </body>
12 </html>

```

- ファイルを「hello.html」と名付けて保存してください。

HTML の実行

- ウェブページの本質は HTML なので、HTML ファイルを実行するには、普段使っているウェブブラウザに直接に入れて、実行するだけです：



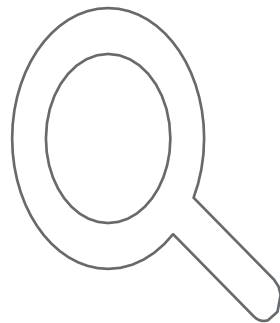
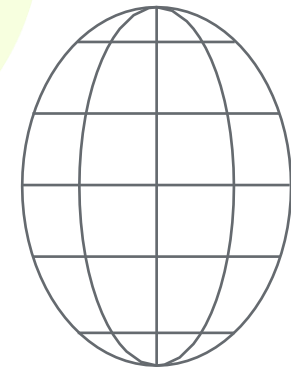
- HTML Living Standard では、一部の古いブラウザでは利用できない機能があるため、当面は**主流なブラウザ（Chrome など）の最新版でテストすることをお勧めします。**

HTML の開発環境

- システム独自のテキストエディタだけで開発すると、構文がハイライトされない、デバッグがしにくいなど、色々な問題があります。
- しかし、一般的に静的ページ開発では、他の言語ほど複雑な開発環境を必要としないため、最初は好みのテキストエディタを選択すればいいです。
- HTML 開発に使用できる代表的なテキストエディタには、Notepad++、Sublime Text、Atom、Visual Studio Code (VS Code) など、さまざまなものがあります。
- 実は、Eclipse 自体でも HTML を開発することができます。



Q&A



目次

- 1 ウェブ開発
- 2 HTML
- 3 **HTML の基本文法**
- 4 代表的なタグ

HTML要素・タグ

- **要素**^[Element]は HTML の最も基本的な構成要素です。
- 各要素は、1 つの**タグ**^[Tag]で定義される。タグは通常、**開始タグ**「<>」、**内容**と**終了タグ**「</>」から構成されます：

```
<h1>This Is A Heading</h1>

<p>This is a paragraph.</p>

<button>This is a button</button>
```

- ただし、内容がなくて終了タグもなく、**開始タグ一つしかない要素**も存在します：

```
<hr>
```

要素のネスティング

- ある HTML 要素の中に、別の HTML 要素を入れることができます。ただし、後者のタグが前者の開始タグと終了タグの間に書かれている必要があります：

```
<html>
```

```
<head>
```

```
<title>Page title</title>
```

```
</head>
```

```
<body>
```

```
<h1>This is a heading</h1>
```

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

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

```
</body>
```

```
</html>
```

HTML ドキュメント構造

HTML Living Standard ドキュメントとして宣言する

1	<code><!DOCTYPE html></code>	<code><html></code> 要素には、ページのすべてのコンテンツが含まれる
2	<code><html></code>	
3	<code><head></code>	<code><head></code> 要素には、ページに関するメタ情報が含まれる
4	<code><title>Hello</title></code>	
5	<code></head></code>	<code><title></code> 要素はページのタイトルを指定
6		<code><body></code> 要素には、ページ上のすべての可視な要素が含まれる
7	<code><body></code>	
8	<code><h1>This Is A Heading</h1></code>	<code><h1></code> 要素は大きな見出しを定義
9		
10	<code><p>This is a paragraph.</p></code>	<code><p></code> 要素は段落を定義する
11		
12	<code><button>This is a button</button></code>	<code><button></code> 要素はボタンを定義する
13	<code></body></code>	
14	<code></html></code>	

属性

- HTML の各要素は、いくつかの**属性**^[Attribute]を持つことができます。属性は、名前、タイプなど、要素に関するいくつかの具体的な情報を提供します。
- 属性は常に開始タグで指定されます：

```
<p 属性名="属性値"></p>
```

- 複数の属性を指定する場合は、空白で区切ります。
- 例えば、先ほどの `<h1>` タグに `title` 属性を追加するには：

```
<h1 title="Hi!">This Is A Heading</h1>
```

This Is A Heading
Hi!

グローバル 属性

- 各要素はそれぞれいくつかの固有の属性を持っていますが、どの要素にも適用できる属性もあります。それらは**グローバル属性**[Global Attribute]と呼ばれます。
- 代表的なグローバル属性は以下の通り：
 - **id** : 要素の ID を定義
 - **class** : 要素のクラス定義
 - **style** : 要素のスタイル定義
 - **lang** : 要素の言語の定義
 - **title** : 要素のタイトルの定義
- id、class、style については、CSS を紹介する際に詳しく説明します。

コメント

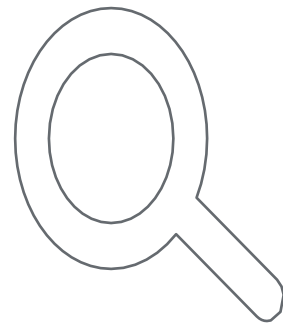
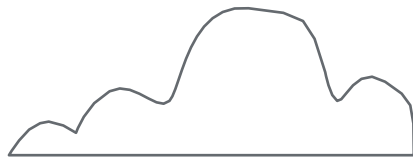
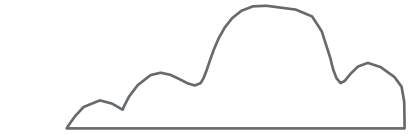
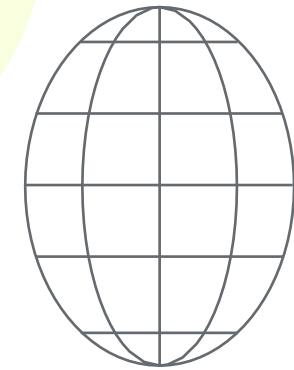
- コメントを書くには特別なタグを使用：

```
<!-- This is a comment. -->
```

- このタグは、ブラウザーに無視されます。タグの中で改行が可能なため、複数行のコメントもこのタグで書くことができます。
- なお、「<!--」と「-->」は、すべての文字が連結され、**スペースを入れてはいけません。**



Q&A



目次

- 1 ウェブ開発
- 2 HTML
- 3 HTML の基本文法
- 4 代表的なタグ

HTML 要素一覧

- すべての HTML 要素とそれに対応するタグは、こちらからご覧できます：

 <https://developer.mozilla.org/en-US/docs/Web/HTML/Element>

- 本日は、最も一般的な要素のみを取り上げます。



参考 : <https://webliker.info/html/46840/>

<head> タグ

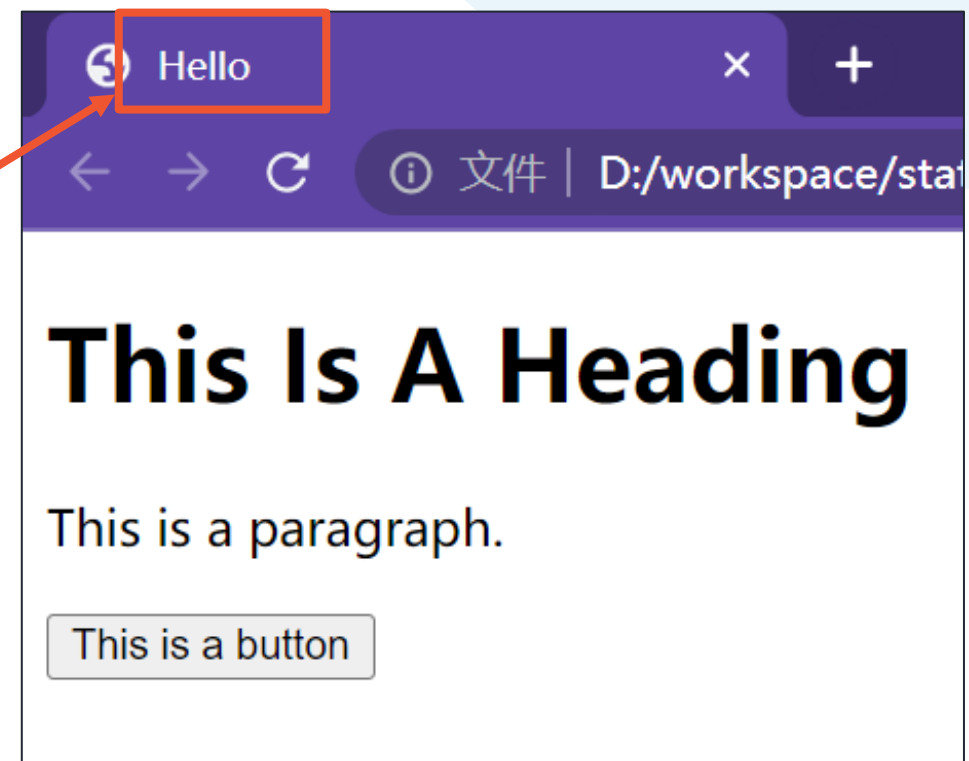
- **<head>** タグはメタデータを保持する要素を記述します。
- **メタデータ** [Metadata] は、「データのデータ」とも呼ばれ、文書の内容とは関係なく、HTML 文書そのものに関わるデータを記録する。例えば、**文書のタイトル**、**文字セット**、**スタイル**（CSS）、**スクリプト**（JavaScript）など。

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

<title> タグ

- **<title>** タグは、ウェブページの**タイトル**を指定するためのものです。このタイトルは、ブラウザでページを開いたときに表示されます。

```
<!DOCTYPE html>
<html>
<head>
  <title>Hello</title>
</head>
```



<link> タグ、 <style> タグ、 <script> タグ

- **<link>** タグは、外部スタイルシート、すなわち外部 CSS ファイルへのリンクに使用されます。
- **<style>** タグは、内部スタイルシートを直接記述する、つまり CSS コードを直接 HTML 文書に埋め込むために使用されます。
- **<script>** タグは、スクリプトを書いたり、外部のスクリプト、すなわち JavaScript のコードを取り込むために使用されます。
- これらのタグについては、CSS と JavaScript を紹介するときに詳しく説明します。

<meta> タグ

- 上記以外のメタデータは、基本的に **<meta>** タグで定義されます。
- <meta> タグの最も重要な機能の一つは、**文字コード指定**：

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

- これに加えて、<meta> タグで、ページの著者、概要、キーワード、テーマ、ビューポートなどのデータを定義することができます。これらのデータは、ページの内容に直接影響を与えるわけではありませんが、ブラウザやページを利用する他のプログラム（クローラー、検索エンジンなど）は、ユーザー体験を向上させるためにこれらの情報を利用することができます。

見出し

- **<h1> ~ <h6>** タグは**見出し**を定義する。<h1> タグは**最大**の見出しで、<h6> タグは**最小**の見出しを定義：

```
<h1>Heading 1</h1>
<h2>Heading 2</h2>
<h3>Heading 3</h3>
<h4>Heading 4</h4>
<h5>Heading 5</h5>
<h6>Heading 6</h6>
```

Heading 1

Heading 2

Heading 3

Heading 4

Heading 5

Heading 6

段落

- **<p>** タグは、**段落**を定義するタグです。段落内容のテキストは、そのままページに表示されます。
- 注意：見出しや段落の中にある改行や、余分な空白文字（連続した空白など）は省略されます。

```
1 <h1>Heading 1</h1>
2 <p>First paragraph.</p>
3 <p>
4   Second paragraph.
5   This is a longer paragraph.
6 </p>
```

Heading 1

First paragraph.

Second paragraph. This is a longer paragraph.

改行

- 改行を加えるには、**
** タグを使います：

```
1 <h1>Heading 1</h1>
2 <p>First paragraph.</p>
3 <p>
4   Second paragraph.<br>This is a longer paragraph.
5 </p>
```

Heading 1

First paragraph.

Second paragraph.
This is a longer paragraph.

テキストのエフェクト

- テキストを以下のタグで囲むと、特別な**エフェクト**が得られます：

：太字

<i>：斜体文字

<mark>：マーク文字

<small>：小さい文字

<ins>：追加文字

<sub>：下付き文字

<sup>：上付き文字

これは**太字**。

これは*斜体文字*。


これはマーク文字。

これは小さい文字。

これは追加文字。

これは_{下付き文字}。

これは^{上付き文字}。

Try  text-effect.
html

HTML エンティティ

- HTML **エンティティ** [Entity]とは、「&」で始まり「;」で終わる特殊な文字列のことです。
- 特定の**特殊文字**を表示したい場合、記号名やその Unicode 番号を使って HTML エンティティを記述することができます。
- 重要なエンティティとして以下のものがあげられます：

エンティティ	文字
 	半角空白
<	「<」
>	「>」
"	「"」
&	「&」

水平線

- `<hr>` タグは、**水平の横線**を引きます：

```
1 <h1>見出し</h1>
2 <hr>
3 <p>段落 1</p>
4 <hr>
5 <p>段落 2</p>
```

見出し

段落 1

段落 2

 タグ

- **** タグは、しばしば特定のテキストの**コンテナ**として使用される。それ自体はコンテンツのテキストに影響を与えるものではありません。
- しかし、CSS を使ってテキストにスタイルを付ける場合、 タグを使えば、どの部分を変更するのかを指定することができます：

```
1 <h1>この部分だけは<span style="color:red">赤い</span>です。</h1>
```

この部分だけは**赤い**です。

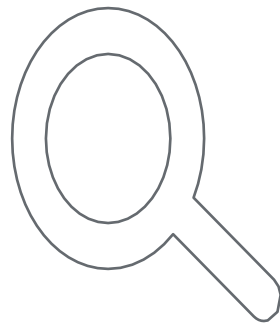
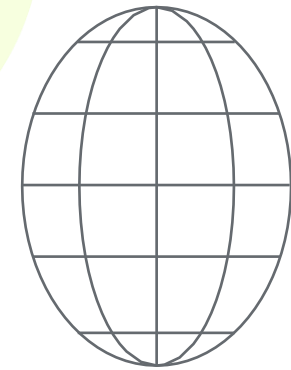
<div> タグ

- **<div>** タグは、他の HTML 要素の**コンテナ**としてよく使われます。
- **** タグと同様、**<div>** 自体は他の要素の内容に影響を与えませんが、いくつかの要素をまとめて、一つの要素として扱うことができます。
- **<div>** タグの主な機能は次の2つ：
 - ページ内の要素**構造を整理**し、コードの可読性を高める。
 - CSS と連動して **<div>** 内の**全要素の統一**的なスタイル指定。

Try 
div.html



Q&A



ハイパーリンク

- **<a>** タグは**ハイパーリンク**を定義します。
- ハイパーリンクは、**ローカル**（ウェブページと同じコンピュータ上）のファイル、または**インターネット上のファイル**にリンクすることができます。
- **<a>** タグの構文は以下のとおりです：


```
<a href="リンク">表示テキスト</a>
```
- **href** 属性にはハイパーリンクがリンクするファイルのアドレス（**URL**）を定義します。タグの内容には、表示されたいテキスト書きます。



ハイパーリンクの target 属性

- ハイパーリンクの **target** 属性は、リンクをクリックするときのファイルを開く場所を指定する。
- target を「**_blank**」に指定すると、元のページを閉じることなく、新しいタブでファイルを開くことができます：

```
<a href="hello.html" target="_blank">別のページ</a>
```

Note



インターネット上のファイルへの URL は「http://」または「https://」で始めます。

画像

- **** タグは**画像**を表示します。
- **src** 属性には画像ファイルの **URL** を指定します。
- **width** と **height** 属性で、画像の**幅**と**高さ**（ピクセル単位）を指定できます。
- **alt** 属性は、画像ファイルが読み込めない場合に表示されるテキストを指定します。
- また、**<a>** タグの中に入れてハイパーリンクが付く画像を作ることにも可能です。



箇条書き

- HTML の**箇条書き**（リスト）には、番号付きと番号なしの2種類があります。
 - `` タグは、**番号なし**箇条書き[Unordered List]を定義。
 - `` タグは、**番号付き**箇条書き[Ordered List]を定義。
 - `` タグはリスト内の**項目**[List Item]を定義。

```
<ul>
  <li>Apple</li>
  <li>Banana</li>
  <li>Cherry</li>
</ul>
```

- Apple
- Banana
- Cherry

```
<ol>
  <li>Apple</li>
  <li>Banana</li>
  <li>Cherry</li>
</ol>
```

1. Apple
2. Banana
3. Cherry

テーブル

- **<table>** タグは、**テーブル**の構造を定義するタグです。
- テーブルの中で、**<tr>** タグでテーブルの**行**を定義する必要があります。
- 行の中で、また **<td>** タグでテーブルの**セル**を定義します。
ここで、**<th>** という特別な種類のセルがあり、**ヘッダーセル**を定義します。

Name	Salary
Tom	\$100
Mark	\$150
Zhang	\$130

Try 01011
11010
01011
table.html

- 注：デフォルトにはこのような境界線がありません。

セルの属性

- セル（<td>、<th>）の **colspan** 属性は、その幅が何列を占めるかを指定します。
- **rowspan** 属性は、その高さが何行を占めるかを指定します。
- それらを組み合わせることで、様々な形のセルを作成可能：

	1	2	3
1	A	B	
2	C	D	
3			

Tips💡

<table> 要素は、表形式のデータを描画するために使用するだけでなく、ページのレイアウトの制御にも役に立ちます。

テーブルのキャプション

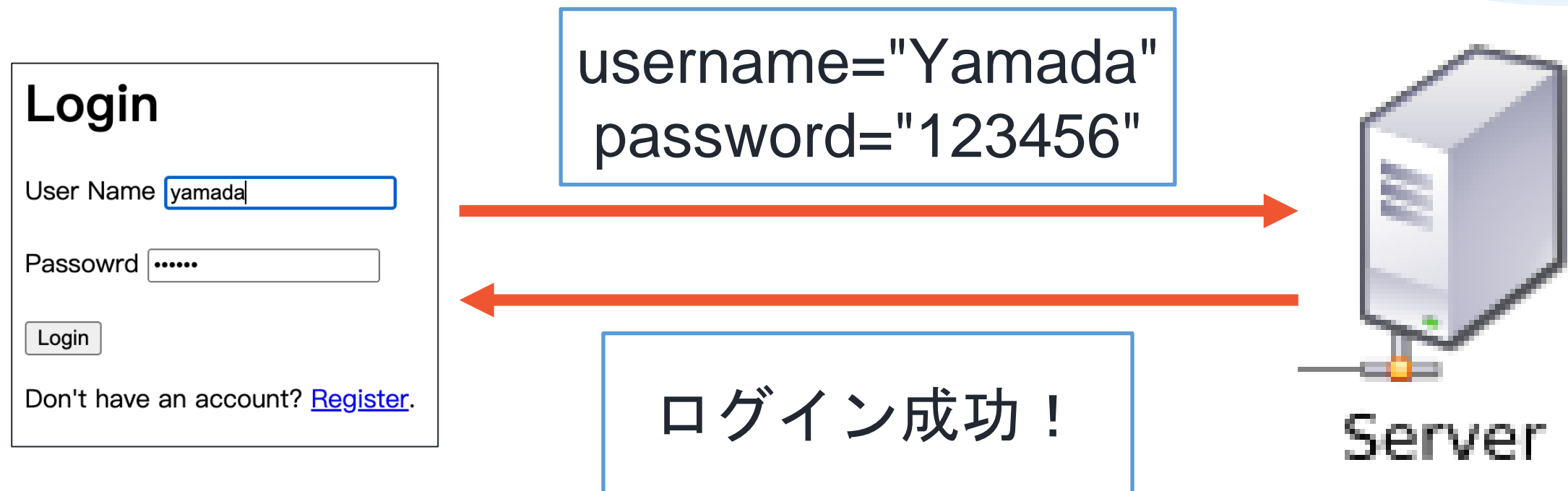
- **<caption>** タグでテーブルの**キャプション**（タイトル）を指定することができます：

色々な
セル

	1	2	3
1	A	B	
2			
3	C	D	

フォーム

- **<form>** タグは、**フォーム**を定義するタグです。
- **フォーム**^[Form]とは、ユーザーの入力を収集し、ユーザーが入力したデータをサーバーに送信して処理するために使用されます：



<input> タグ

- フォームの中で、<input> タグで特定の**ユーザー入力**を定義します。
- <input> タグの **type** 属性を設定することで、様々な**入力タイプ**を表示することができる。例えば：

タイプ名	説明
text	普通のテキスト
password	パスワード
radio	ラジオボタン
checkbox	チェックボックス
file	ファイル
submit	送信ボタン

<input> の属性

- <input> タグの共通な属性は次の通りです：

属性	意味
id	ID（HTML 中の名前）
name	名前（サーバーが使用する名前）
type	入力タイプ
value	デフォルト値
placeholder	入力されていない時に表示されるテキスト
required	入力が必要かどうか
pattern	満たすべきパターン（正規表現で書く）

<label> タグ

- フォームの **<label>** タグは、入力に対応した**ラベル**のテキストを定義する。
- <label> タグの **for** 属性を使って、どのラベルがどの入力に対応するかを指定することができます。 **<input> の ID と <label> の for** を同じ値にする必要があります。


```
<label for="name" >ユーザー名: </label>
<input id="name" type="text">
```

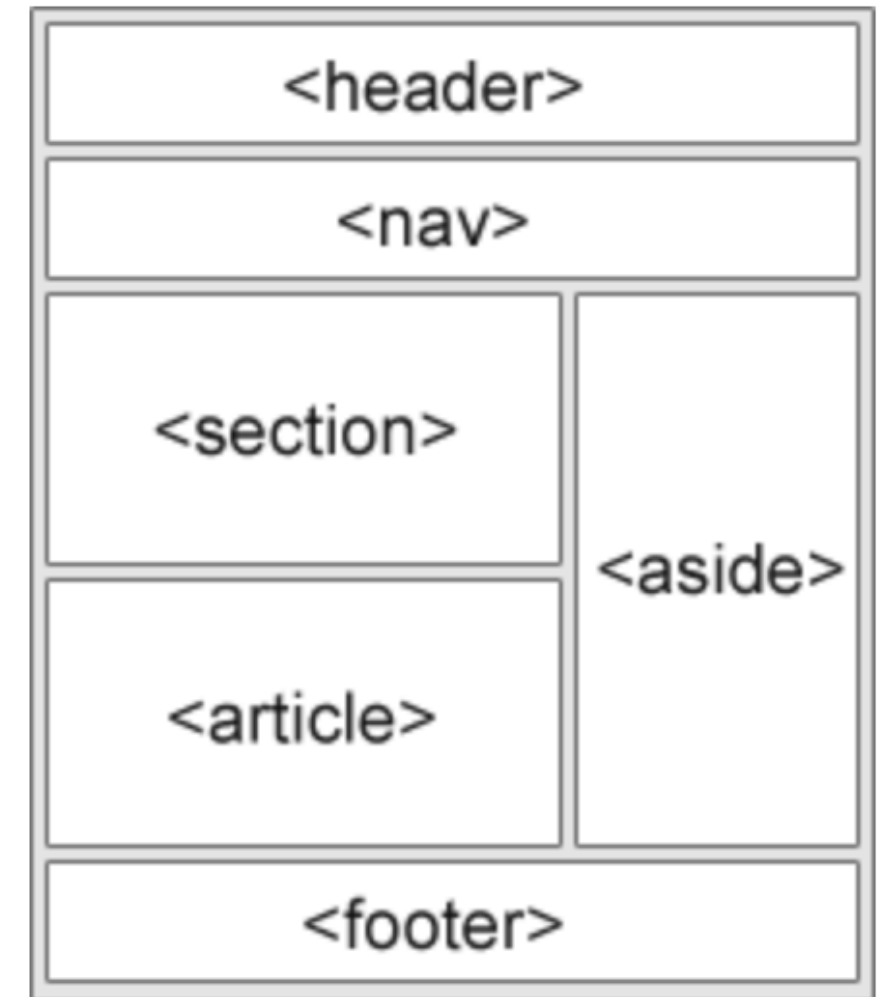
Try 01011
11010
01011
form.html

<form> の action 属性と method 属性

- **action** 属性は、フォームが送信されたときに HTTP リクエストの**ターゲット URL** を定義します。ユーザーが送信ボタンをクリックした後、ユーザーの入力データは URL で指定されたプログラム（例えば、サーバサイドの Java プログラム）に送信されます。
- **method** 属性は、HTTP リクエストの**メソッド**を定義する。主に GET と POST の 2 つのメソッドが使用されます。
 - **GET メソッド**で送るデータは誰でも見えて、少量のデータしか転送できません。
 - フォームのデータに、セキュリティ情報や個人情報が含まれている場合、またはフォームに大きなデータ（ファイルなど）が含まれている場合は、**POST メソッド**を使用するべきです。

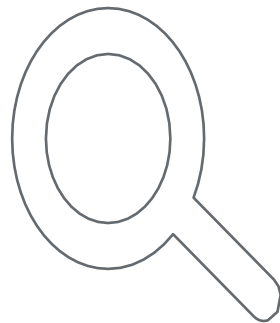
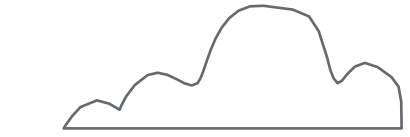
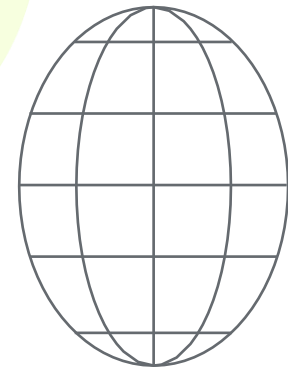
HTML セマンティック要素

- <div> 要素で他の要素をまとめられますが、どの部分がどのような役割があるのがわからなくなります。**役割**を明確に示すためには、**セマンティック要素**[Semantic Element]を使います：
- セマンティック要素は構文的に通常の <div> 要素と変わりませんが、ブラウザや他のプログラムは、それらの役割に対応してユーザーの体験を向上させることができます。
- 全部のセマンティック要素はここでチェックできます：
 <https://developer.mozilla.org/en-US/docs/Glossary/Semantics>





Q&A



レスポンスデザイン

- **レスポンスデザイン** [Reponsive Design] とは、あらゆるデバイスで正常に表示されるようなウェブページの作成を意味します。
- レスポンスデザインでは、さまざまな**スクリーンサイズ**に対応できる要素の表示法が求められます。
- レスポンスデザインの最初のステップは、適切な**ビューポート** [Viewport] を設定することです。ビューポートは **<meta>** タグで設定することができます：

```
<meta name="viewport" content="width=device-width, initial-scale=1.0">
```

ビューポート

- ビューポートはページ作成時点でブラウザにページのサイズを伝えて、ページの歪みを防止できます：



ビューポート設定なし



ビューポート設定した

URL (1)

- **URL** (Uniform Resource Locators) とは、コンピュータやインターネット上のあらゆるファイルのある場所を表す文字列です。普段「ウェブアドレス」と呼ばれるものも、URL であります。
- もう一つ、**URI** (Uniform Resource Identifier) という概念もありますが、これはもっと広い範囲のファイルの識別子で、URL はこれに含まれます。意味的には大差ないので、混用されることもしばしばあります。

URL (2)

- URL は、一般的に次のような構文になっています。

```
scheme://prefix.domain:port/path/filename
```

- **scheme** は、**通信の仕方**を定義（最も一般的なのは「http」または「https」で、HTTP 通信を意味する）
- **prefix** は**ドメイン**[Domain]のホスト[Host]名（HTTP のデフォルトは「www」）
- **domain** は**ドメイン名**を定義（「google.com」とか）
- **port** はホストの**ポート**[Port]番号（HTTP のデフォルトは 80）
- **path** はサーバ上の**パス**（ディレクトリ）
- **filename** は**ファイル名**（「index.html」とか）

まとめ

Sum Up



1. ウェブ開発の基本概念：

- ① サーバー、クライアント、HTTP リクエスト。
- ② HTML、CSS、JavaScript の関係。

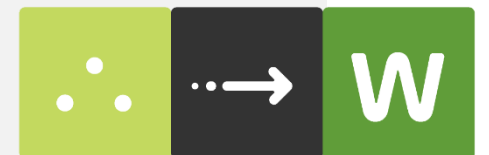
2. HTML の基本的な構文である要素、タグ、属性。

3. HTML の常用タグ：

- ① `<head>` タグとメタデータ。
- ② テキスト、画像、ハイパーリンクなどの基本タグ。
- ③ テーブル、箇条書き、フォームなどのコンテナ。

Thank you!

From Seeds to Woodland — Shape Your Future.



Shape Your Future