情報デザイン演習II 3.HTML実践

- Chapter1 最初に知っておこう!Webサイトの基本
- 旧:Webデザインの基礎知識
- Chapter2 Webの基本構造を作る!HTMLの基本
- 画像の種類と特性について
- 画像の加工
    - Photoshop
    - XDでの書き出し
- データの位置
    - 他のページにリンクする指定
    - 同じページ内のリンクを指定
- 演習

先週のおさらい

- HTMLの歴史
- HTMLの基礎
  - HTML5/HTML Living Standard
  - Visual Studio Code使い方おさらい
  - 最低限のHTML
  - HTMLタグ(開始タグ・終了タグ)
  - インデント
  - h1〜h6, ul,ol,li, dl,dt,dd, img, a, br, hr
  - コメントアウト
  - チートシート

今日やること

テキストをみんな持ってきていると思うので、

- Chapter1 最初に知っておこう!Webサイトの基本(P.14-P.48)
- Chapter2 Webの基本構造を作る!HTMLの基本(P.50-P.86)

までやっていこうと思います。前回までの内容をより詳しくやっていきます。
2回一応説明しているので、適当に飛ばしながら行きます。

教科書に書いてなくても知ってて欲しいことは都度触れようと思います。

まずは、先週の課題について軽く確認してみましょう。

Chapter1 最初に知っておこう!Webサイトの基本

よいWebデザインとは?

> デザインを通して「何を伝えたいか」ということが大切

> 使いやすいことが重要

さまざまな種類のWebサイト

- コーポレートサイト
- プロモーションサイト
- ポートフォリオサイト
- ショッピングサイト
- メディアサイト
- SNS

ユーザビリティとは

> ユーザビリティ = 「使いやすさ」

見やすいデザインにする

- 色使いに注意する
- 目立たせたいものを明確にする
- レイアウトを統一

> (参考)デザインの4原則

読みやすい文章にする

- 結論を先に書く
- 専門用語を使わない
- 簡潔にまとめる

ユーザビリティとは

使いやすい操作性にする

- 予想ができる様にする
- 動作を速くする
- 一目見てわかる様にする

Webサイトの仕組み

- インターネットとは
- Webとは
- Webページの仕組み(サーバ・」クライアント)
- URLとは

デバイスの種類

端末

- iOSデバイス
- Androidデバイス
- モバイルデバイス
- スマートデバイス
- ウェアラブルデバイス
- IoTデバイス

ん？デスクトップパソコンが分類されていない気が...

デバイスの種類

周辺機器

- USBデバイス
- ストレージデバイス
- オーディオデバイス

ここにこの記載なくても構わないような...

ブラウザの種類

- Google Chrome
- Safari
- Microsoft Edge
- Firefox

シェア

第一版には

- Microsoft Internet Explorer

が記載されていましたが、ついになくなりました！！！

- Microsoft 社 Internet Explorer のサポート終了について
- 脱線：IEの為のCSSハック

レンダリングエンジン

表示を担当するソフトウェア。レンダリングエンジンが異なると違う表示をする可能性が増える。

各Webブラウザが使っているレンダリングエンジン

Javascriptエンジン

Javascriptの実行を担当するソフトウェア。レンダリングエンジンが異なると異なる挙動をする可能性がある。

ブラウザとJavaScriptエンジンの関係とJavaScriptエンジンの構造

制作の流れ

1. 企画を立てる
2. サイトマップを作る
3. ワイヤーフレームを作る
4. デザインする
5. コーディングする
6. Web上に公開する

制作を始める前に

- テキストエディタ(VSCode)をインストールする
- VSCodeを日本語化する
- ブラウザをインストールする(Chrome)

グラフィックツールを確認する

- XD(おっと、本から消されてる...)
- Figma
- Photoshop
- Illustrator

旧:Webデザインの基礎知識

現テキストが触れていない(第二版で削除された)必要なところだけ紹介しておきます。

旧:Chapter1-01 インターネットとWebの歴史

HTMLの仕様の確認方法

- W3C(World Wlde Web Consortium)と呼ばれる団体がHTMLを策定していた。
- WHATWG(Web HyperText Application Technology WG)というApple, Mozilla, Operaの開発者らが設立した団体ができた。
- 一時は分裂状態になり、HTMLが2パターンありそうな状態になる。
- 現在では、W3C による HTML の仕様策定は中止され、WHATWGが策定を進めるHTML Living Standard が HTML の標準仕様となっている。 - (いろいろあったものの、協力関係は今後も続いていく)

HTML Living Standard
HTML Living Standard(日本語訳)

旧:Chapter1-02 Webサイトの種類

Webサイトを規模で分ける

- シングルページサイト：1ページで完結
- 軽量サイト：数ページ程度
- 階層構造を持つサイト
- 大規模サイト

Webサイト構築と情報デザイン

- Webサイトは、紙媒体の「ページ」と大きく異なる
- ナビゲーションメニューなどのUI(ユーザーインターフェイス)を含む

よって

- 「みやすく・読みやすい」魅力的なページデザイン
- ユーザに対して「使いやすく・アクセスしやすい」サイト設計

この二つが求められる。

Webサイト制作の3つの考え方

「古い環境はどうするのか？」はWeb制作に関して悩ましい問題。

テキストでは3つの用語をあげているが、とりあえず

- 仕事でWebデザインする場合には、OS, ブラウザ条件を決めること

が必須となる。

旧:Chapter1-03 Webブラウザーの使い方

Webブラウザーの「先行実装」について

HTMLはWHATWGにて策定されているのは先ほど説明した通り。CSSはW3Cにて策定されています。

策定のプロセスは「こんなのどう？」「これでいこう」と変わっていくわけですが、実際にブラウザーを作成する側は、「こんなのどう？」と言っている時に、「とりあえず実装してみた」としていたりします。

このように、仕様が確定する前に先に実装することを「先行実装」と言います。

そうすると、このブラウザーでは使えるのにあのブラウザーでは使えない、ということが起こります。

不便ですので、この状況をまとめているサイトがあります。

Can I Use...

ベンダープレフィックス

先行実装する際には、固有の文字を付けることになっています。
そのため、仕様が確定するまでは、4つくらい同じ命令を書く必要があります。

仕様が確定されてから使えばいいのに、という考え方もありますが、

- これって本当に必要だよね

と色々な人が試した上で認定されてからしか仕様確定に至らないので、一応知っておいてください。

- 今さら聞けない！ベンダープレフィックスとは【CSS】

Chapter2 Webの基本構造を作る!HTMLの基本

HTMLとは

> HyperText Markup Language

Webページを作る上で土台となる言語

> タグ：「<」「>」で挟まれた文字列

HTMLファイルを作ろう

今日の作業フォルダ「ID_03」を「ID_root」内に作成してファイル名を「sample2-2.html」として以下の内容を書いてみましょう。

```
<!DOCTYPE html>
<html lang="ja">
<head>
    <meta charset="UTF-8">
    <title>猫の実態</title>
    <meta name="description" content="猫の好きなもの、日々の生活をご紹介">
</head>
<body>
    <h1>猫の1日</h1>
    <p>ひたすら寝ています。</p>
</body>
</html>
```

HTMLファイルを作ろう

- LiveServerで確認しましょう。
- VSCのエクスプローラーで「ID_root」が表示されているかな？
- インデントは正しくついているかな(head,bodyはインデントなしでOKです)

HTMLファイルを作ろう

ファイル名について

- 拡張子はファイルの種類を表す文字列「.html」「.css」など必ずつけよう
- 「.htm」とすることも一応あるが、「.html」にしよう
- 記号は「-」「_」以外は使わない
- 空白(全角・半角とも)は利用しない
- 小文字に統一する

特殊なファイル名

- index.htmlはディレクトリにアクセスした時に表示される特殊なファイル名
- default.htmlもindexがない時に使われる。

HTMLファイルの骨組み

- <!doctype html> HTML5(HTML Living Standard)で書かれていることを宣言している
- <html>〜</html> HTMLの文書を表している。「ja」は日本語の略
- <head>〜</head> ページの情報を記述(表示されない)
- <body>〜</body> ページの本文を記述
- <meta charset="UTF-8"> 文字コードがUTF-8であることを宣言
- <title>〜</title> ウィンドウやタブのタイトルに表示される
- <meta name="description" content="〜"> ページについての説明文。検索エンジンが利用する。
- htmlタグ内には必ずheadタグ,bodyタグが入る

参考：metaタグについて 基本、検索エンジンやSNSが利用する。

HTMLの基本の書き方を身につけよう

HTMLの基本文法とタグ

- 開始タグ・終了タグで囲むのが原則
- br, hr, img, input, link, metaタグ等は終了タグがいらない

HTMLが正しく記載されているかのチェックサービスが一応あります。
なるべくエラーを出さない様に記述するのが望ましいです。(厳密なのでエラー無しにするのはかなり難しい)

- WHATWG HTML Conformance Checkers

たまに不具合出るので他のリンクも貼っておきます。

- HTMLコードのチェックが簡単にできるおすすめツール5選

HTMLの基本の書き方を身につけよう

- 半角英数字で書く
- 小文字で統一しよう
- 入れ子を意識しよう
- タグに情報を書き加えることができる(属性)

見出しをつけよう・文章を表示しよう

- 見出し：h1〜h6(飛ばさない)
- 段落：p
- 改行：br(閉じタグなし)

コンテンツモデル(テキスト外)

> 〇〇要素の中には、△△要素が配置できる」といった、各要素ごとに中に配置できる要素を定義した配置ルール

があります。

ちょっと難しいので、動画を紹介します。こんな考え方あるのか、くらいでいいです。

- 要素の配置ルールとカテゴリー(15:57)

画像を挿入しよう

後回しにします。

リンクを貼ろう

- aタグで囲む
- 画像でも、文章でもOK
- とすることで別タブで開く

リストを表示しよう

- ul: 順序なしリスト unordered list
- ol: 順序付きリスト ordered list
- li: リスト項目 list item

表を作ろう

これは少しやってみましょうか。
「table.html」を作成して
c2-10-1, c2-10-2, c2-10-3, c2-10-4
をやってみましょう。(こんな感じで作れるってのが分かれば十分です)

table, tr, th, tdが基本ですが、
thead, tbody, tfootというタグも使える様になっています。

- 参考：【HTML】表の構造を明確化するthead/ tbody / tfoot要素 とは？わかりやすく解説！

フォームを作ろう

これもやってみましょうか。サーバ側を書かないと何も動きませんが…
「form.html」を作成して

- c2-11-1 〜 c2-11-10
- c2-12-1

を入力してみましょう。(こんな感じで作れるってのが分かれば十分です)

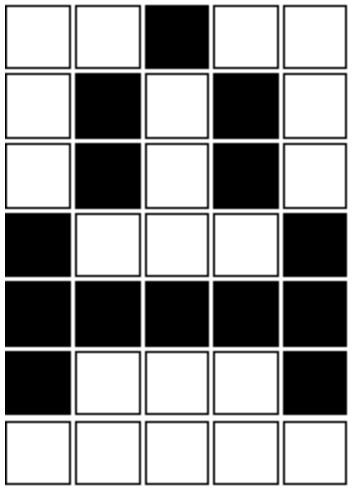
画像の種類と特性について

ラスターグラフィックスの特性

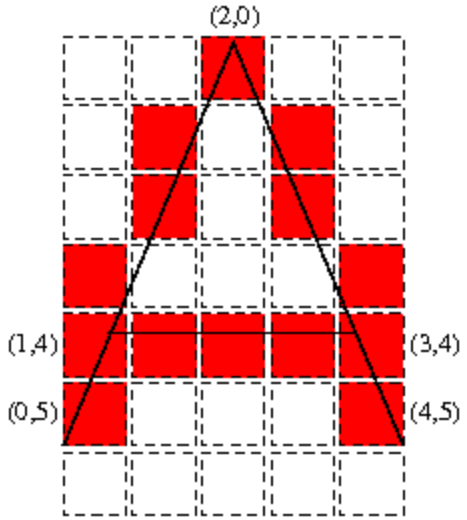
- ラスターグラフィックス(ビットマップ)
- ベクターグラフィックス

単純に言えば、Photoshopで扱う写真は前者、Illustratorのパスで扱う画像が後者、ということになります。

ラスター画像では解像度について考える必要があります。
解像度が高ければ綺麗ですが、ファイル容量が大きくなってしまいます。最適な解像度にする必要があります。



Bitmap–Darstellung des
Buchstaben "A"



Vektor–Darstellung
des Buchstaben "A"
(mit unterlegter Bitmap–
darstellung)

可逆圧縮・非可逆圧縮

写真のデータは重いため、圧縮してWeb上で利用されます。次の二通りあります。

可逆圧縮

圧縮する前と同等な品質で閲覧可能です。

非可逆圧縮

元画像と比べると、品質が劣化します。その代わり、画像サイズを小さくできます。

背景透過

普通の画像は長方形になりますが、背景透過を使うことで、形状を変更することが可能となります。

形式の使い分け

画像ファイルの形式によって使い分けが必要となってきます。
それぞれの特徴をよく理解しておきましょう。

GIF

- 256色以下
- 背景を透過することができる
- アニメーションも可能
- 拡張子は.gif

画質は良くなく、グラデーションが苦手だが、ロゴなどではファイルサイズを小さくすることができるため利用された。

JPEG

- 非可逆圧縮
- フルカラーに対応
- 背景透過できない
- 拡張子は.jpg または.jpeg

写真では今でも良く利用される。

PNG-8

- 256色以下
- 背景を透過することができる
- 拡張子は.png

GIFより軽くなりやすい

PNG-24

- 可逆圧縮
- フルカラーに対応
- 背景透過もできる(PNG-32)
- 拡張子は.png

SVG

- ベクター形式
- 拡張子は.svg

ロゴなどに向く。

豆知識：svgファイルは実はテキストファイル...

WebP

- 非可逆圧縮/可逆圧縮
- 背景を透過することができる
- アニメーションも可能
- 圧縮率が高い
- 拡張子は.webp

他の形式と比べて後からGoogleが開発したフォーマットのため、高性能ですが、ブラウザ対応が進まずなかなか普及しませんでした。Photoshopも標準で対応するようになったので、そろそろ使われてきています。

AVIF

まだあまり普及していないイメージがあります。性能はいいんですが...

- 非可逆圧縮の圧縮率が高い
- 可逆圧縮／非可逆圧縮を選択できる
- GIFのようなアニメーションを制作できる
- 透過処理ができる
- HDRに対応している
- AVIFとは？特徴や注意点から変換できるアプリ・ツールまで紹介

Photoshopでの書き出し

PNG,JPG,GIFは仕上がりを見ながら品質を変えられるんですが、
WebP,AVIFはまだそれができません。

無理せず、PNGやJPGでもまだいいかな、という気もします。

画像フォーマットの比較

- PNG でまだ大丈夫？　JPEG, WebP, AVIF の容量と画質を徹底比較！

HEIF(番外)

最近のiOSで撮影すると画像はHEIFという形式になっています。

これはWebで利用できないため(Safariは表示可能になっている)、変換する必要があります。

画像の加工

サイズ・容量に気をつけよう

- 画像の容量が大きいと、表示される時にそれだけ時間がかかります。なるべく小さくしましょう。
- 小さいのは良いことですが、小さくし過ぎると汚くなります。
- 容量には以下の点が非常に関係しています。
    - 縦横のピクセル数
    - 圧縮形式

縦横のピクセル数の変更には

- イメージ - 画像解像度
- イメージ - カンバスサイズ

をうまく使いましょう。

ファイル劣化の見方

圧縮し過ぎると

- グラデーションが汚くなる
- 線の周りにブロックノイズ(正方形のノイズ)が出る

となります。気をつけましょう。

Photoshop

webpの書き出し

別名で保存をして書き出すことができますが、どのくらい汚くなるかを現状書き出し時に確認できないのが面倒です。

これがまだwebpをそこまで推奨できない原因となります。

gif, jpg, pngの書き出し

個人的には、古いのですが

- ファイル - 書き出し - Web用に保存(従来)

が好きです。プレビューがみやすいからです。ただ、現在では

- ファイル - 書き出し - 書き出し形式

を使うのが普通だと思います。

画像を書き出してみよう。

1. FREEPIKから好きなものをダウンロード
2. zipを解凍してpsdファイルをPhotoshopで開く
3. 書き出したい画像をコピー(Command+C)
4. 新規作成でクリップボード(Command+N) 透過させたい場合にはカンバスカラーを透明に
5. ペースト(Command+V)
6. ファイル-書き出し形式、「書き出し」

あ！2分割で確認できるようになってる…
JPGにして、画質を変えて、劣化するのを確認しましょう。

Figma

前期も使ったと思うんだけど...

- 【完全初心者ガイド】Figmaブラウザ版とアプリ版を徹底比較！使いやすいのはどっち？

どちらでも構いません。

Figmaでの書き出し やってみよう

自分まだFigma真剣に触ってないので詳しくはここを参照のこと。

- 【Figma入門⑦】画像の書き出し・エクスポートの設定方法を解説

ここからダウンロードして、画像をいくつか書き出してみましょう。

- 無料で使えるFigmaテンプレート35選。Webデザイン/モバイルUI/デザインガイドライン作成におすすめ

1. 選択して(グループとか気をつけて中に入らないとかもですね)
2. 右のパネルの一番下のエクスポートからエクスポート

で終わり。簡単！

データの位置

データの位置について

前回画像を表示してみましたが、ネット上の画像だったためにURLを貼ればOKでした。
データの位置を表現するには

- URL
- 相対パス
- 絶対パス

と方法があります。

これは、画像だけでなく、ファイルの位置を指定する時にも利用します。

URL

詳しくはやりませんが、

Uniform Resource Locatorと言って、インターネット上のリソースを特定するための形式的な表示方法です

- http://(ドメイン名)/(場所)
- https://(ドメイン名)/(場所)
  となります。

ドメイン名は、インターネット上のコンピュータを表す住所のことです。

相対パス

同じコンピュータ内のリソースを利用する時に、自分自身を基準にして、相対的にどこにあるか？として指示する方法です。

index.htmlとtest.jpgが同じフォルダの中にある場合には

```
test.jpg
./test.jpg
```

と書きます。index.htmlと同じフォルダに「img」フォルダがあり、その中にtest.jpgがある場合には

```
img/test.jpg
```

と書きます。

相対パス 特殊な記号

```
./
```

とすると自分自身が属するフォルダを差します。

```
../
```

とすると、一つ上のフォルダを指します。

絶対パス

サーバ等の基準となるフォルダ(ルートと呼ぶ)からの位置を指します。

VSCodeで例えば、
ID_root - ID_03 - img - test.jpg
という位置を表すには

```
/ID_03/img/test.jpg
```

と書きます。

相対パス?絶対パス?どっちを使う？

- サイト全体で利用するロゴなどは「/img」に入れておいて絶対パスで参照すると良いでしょう。
- 自分のサーバ内の位置と近い場所なら相対パスで参照すると良いでしょう。
- サイト外のデータはURLを使うしかないですね。

https://sammyppr.github.io/2025/InformationDesignII/id_03_root.zip

にサンプルを置いたので、ダウンロードしてみてみましょう。

VSCで新しくウィンドウを開いて、解凍したid_03_rootを開いてLiveServerで見てみましょう。

画像の幅と高さを指定

HTMLで強引に幅と高さを指定する方法がありますが、もう使いません。

理由としては、画像の幅・高さの指定は、デザインの領域にあたるため、CSSで指定する方が今の時代としては適切だからです。

見出しを画像で表示する

どうしても特殊なフォントを使いたいなどの場合、見出しを画像で表示します。alt属性は必須ですね。

```
<h1><img src="title.png" alt="インターネット入門"></h1>
```

CSSでデザインがある程度できるようになったため、そんなに使われていないと思います。

他のページにリンクする指定

画像の時と同様、リンク先も自分から見て、どこにそのHTMLファイルが存在するか、ということが重要になります。

ハイパーリンクの指定方法

前回やりましたね

```
<a href="リンク先の場所">リンクを貼る要素</a>
```

こちらも次の3種類が利用できます。

- URL
- 絶対パス
- 相対パス

外部のページをリンクする。

自分のサイトの中で移動する場合には問題ないですが、外部のホームページに飛んでしまうと戻ってきてくれないかもしれません。そのため、別ウィンドウ・別タブで開きます。

```
<a href="URL" target="_blank">リンクを貼る要素</a>
```

テキストリンクの色

これはデザインのためCSSの範囲となります。

同じページ内のリンクを指定

外部のページでなく、同じページの異なる場所、つまり下の方にリンクする時について学びます。

id属性

HTMLタグにはid属性というものを付けることができます。例えば

```
<h3 id="about">私たちについて</h3>
```

となります。idは同じページで同じ名前を付けることはできません。日本語も使いません。

ページ内リンク

そして、ページ内でリンクするときには

```
<a href="#about">私たちについて<a>
```

とすることによって、飛ぶことができるようになります。

HTMLページ以外へのリンク

これまでリンクの飛び先にはHTMLページを指定してきましたが

- PDF
- Zip

など、インターネット上にあるものであれば何にしても構いません。

メーラーの起動

最近では減りましたが、問合せなどでメールを起動することがあります。

URLは実は

- http://www.google.co.jp
- https://www.google.co.jp

の他に

- mailto:someone@someone.com

のようなものもあります。これにより、メーラーを起動することができますが、
あくまで、メールアプリケーションを起動するため、Webメールを起動することはできません。

なお、メーラーの起動でなく、フォームによる問い合わせが一般的なものとなっています。

これでChapter 2まで終わったね

> 2-13 ブロック要素でグループ分けをしよう

は、レイアウトを組むCSSと密接に絡んでいるので、そのタイミングでやりましょう。

演習

- 「ID_03」の中に「01」「img」、「01」の中に「02」「img」、「02」の中に「03」「img」フォルダを作ろう
- 「01」の中に「01.html」、「02」の中に「02.html」、「03」の中に「03.html」のファイルをそれぞれ作ろう
- 3つあるimgフォルダの中に、別の画像を3つ配置しよう
- 01.html,02.html,03.htmlのそれぞれのファイルで3つの画像を表示しよう
- 01.html,02.html,03.htmlに移動できるようにリンクを貼ろう

ID_03をNASに提出してください。