

2021年8月21日（土）

第1回勉強会

今日の題材

- VScodeの基本と応用を知っとく
- ぐちゃぐちゃなコードを綺麗にしよう
- CSSをゴっつ簡単に綺麗に書こうや
- VScodeでサクサクコーディングしようや
- リーダブルコードってなんや？
- CSSのTips
- Sassってよく聞くけどなんや？
- PHPファイルでWordPressの関数がエラー表示されてウザい

VScodeの基本やな

まずは拡張機能をガンガン入れて快適かつ素早いコーディングができるようになるう

拡張機能や、機能を使いこなせばタイプミスやエラーも発生しづらくなり結果的に速さに繋がります

次スライドで最低限必要な拡張機能を紹介します

必須の拡張機能やな

まずは以下の拡張機能を入れよう

Auto Close Tag	Auto Rename Tag
Bracket Pair Colorizer2	Code Spell Checker
Live Preview	Path Intellisense
zenkaku	Autoprefixer
SCSS Everywhere	htmltagwrap

各プラグインの説明をします

PHPファイルでWordPressの関数が エラー表示されてウザい

原因はこいつ → PHP Intelephense

1分で設定できるのでPHP書く人は一緒にサクッと直しましょう

自動フォーマットしようや

汚いコード（読みにくい、インデントが揃っていない
etc...）を書くのはやめよう

VScodeには標準でコードを綺麗に揃えてくれる機能があります
まずは設定を変更しましょう



ショートカットでフォーマットしよう


Mac 「option + shift + F」

Windows 「Alt + shift + F」

VScode応用編やで

次スライドからは応用編です

VScodeを利用して素早く正確なコーディングを行いましょう



断言しよう…
君たちのCSS
は汚い!!

CSSの自動フォーマットもしようや

CSSの自動フォーマットは本当はstylelintというnode.jsプラグインを使うのが一般的且つ使いやすいのですが今回はVScodeの拡張機能であるCSSCombを使用します

CSSCombの設定をしよう



ファイルの保存時に自動でフォーマットするようにしよう



ベンダープレフィックスの説明

余談：*eslint*

タイポとエラーの確認がしたいんや

HTML、CSS、JavaScript等のコードに記述ミスやタイプミスがあると、コンピューターやブラウザは正確にプログラムを処理することができません

ではVScodeで記述ミスやタイプミスを発見するにはどうすればいいでしょうか？

余談：空白の削除設定と全角空白の見つけ方

学校ではなぜか教えてくれへん 素早いコーディングのTips

ここまでくればタイポや記述ミスはほとんど無くなりましたね
では正確なコードを素早く書いていきましょう

HTMLファイルを作成した時、最初に記述するDOCTYPE宣言ですが
手打ちしてる人は今すぐやめましょう

DOCTYPE宣言は1秒で全部打てます！

まずはVScodeの設定を少し変えます

Emmetの使い方

VScodeに限らず大体のコーディング用エディターやIDE（統合開発環境）にはEmmetというコードを高速に打つ機能が搭載されています

※Sublime Textはプラグイン必須

ではDOCTYPE宣言を1秒で打ってみましょう

「!」をタイプしたあとにTabキーを押してみてください

次はもっとEmmetを使ってみましょう

リーダブルコードってのを知っとうや

リーダブルコードとは

より良いコードを書くためのシンプルで実戦的なテクニック

- コードは理解しやすくなくてはならない
- コードは他人が最短時間で理解できるようにかかなければならない
- 明確な単語を選ぶ
- 具体的な名前を使って物事を詳細に説明する
- 大文字やアンダースコアに意味を付与する

余談：DeepLのすゝめ

皆が知りたいであろうCSSのTipsを紹介する で

Tips

- ブロックレベル要素、インライン要素、インラインブロック要素
- 要素を中央に揃える
- 脳死でvwとvh使ってない？
- なんか変な隙間できんねんけど
- rem、em、px、% どれ使えばええの？
- font-size: 62.5%; は本当に万能なのか

ブロックレベル要素、インライン要素、 インラインブロック要素

	width / height	横幅の 初期値	高さの 初期値	他の要素と の並び	余白
ブロック	使える	親要素の 横幅	内容で決 まる	改行される	勝手につくもの がある
インライン	使えない	内容で決 まる	内容で決 まる	改行されな い	上下に特殊な付 き方をする
インライン ブロック	使える	内容で決 まる	内容で決 まる	改行されな い	勝手につくもの がある

要素を中央に揃えよう

悩む人が多いと思うのでブロック要素、インライン要素、インラインブロック要素それぞれのやり方とベストプラクティスを紹介します

まずは左右の中央揃え

インライン要素

```
text-align: center;
```

ブロック要素

```
margin: 0 auto; ※インライン要素の中のブロック要素に注意
```

インラインブロック要素

```
text-align: center; ※親要素に
```


上下の中央揃え

インライン要素

```
vertical-align: middle;
```

ブロック要素

```
position: absolute;
```

```
display: flex;
```

```
inset
```

モーダル表示のTips

君たち脳死でmin-height: 100vh;使ってない？

iOSのSafariにheight: 100vh;とするとアドレスバーの高さ分ずれる
JavaScriptを使って高さを計算するのがベストプラクティスですが今回はCSSだけで実装してみましょう

```
body {  
  min-height: 100vh;  
  min-height: -webkit-fill-available;  
}  
  
html {  
  height: -webkit-fill-available;  
}
```

インラインブロック要素で横並びにすると 変な隙間ができる

きれいなコードを書いたが為に起きてしまう事件です
HTMLにコメントを入れるという荒業もありますがCSSで解決しましょう

```
ul {  
  letter-spacing: -0.5em;  
}  
  
li {  
  display: inline-block;  
  letter-spacing: normal; /* 必須!!! */  
}
```

rem、em、px、% どれ使う？

※まずは各単位の解説をします

ベストプラクティス

- 横幅 → 全体的なレイアウトには%、ブラウザの幅基準ならvw
- フォントサイズ → rem、em
- `border-width` など文字サイズが変わっても変化させたくない → px
- 文字サイズとともに変化させたい余白など → rem、em
- メディアクエリ → rem、em

※全てpx指定のWebサイトも非常に多くコストとの相談になる

font-size: 62.5%; は本当に万能なのか

メリット：

- 1rem = 10pxになるため計算が楽
- ブラウザのフォントサイズ設定を変更したときに、ちゃんとその設定になる
- なんかモダンな感じがする

デメリット：

- chromeは最小フォントサイズが10pxなので
- ブラウザのフォントサイズ指定によっては不都合がある

Sassってよう聞くけどなんなんや

SassはCSSのメタ言語です

変数や関数などが使えて便利

ネストで記述できるのでタイプ量も減るし何より見やすい

コーダーやエンジニア希望の人は知っておきましょう

書けなくても知ってるだけで印象が違います

今回時間が足りないのでサクッと使ってみます

詳しく知りたい方は個人的にお声がけください

余談：Macの方向けの話

便利なソフト使ってる？

コーディングとは直接関係ありませんがMacを使っている方向けに便利なソフトを紹介します

- Run Cat
- Alfred
- ImageOptim
- クリップボード拡張