FAINPIXAR

コーディングガイドライン

Ver.1.5.0

作成日：2015/02/10
更新日：2020/06/19

FAINPIXAR

目次

FAINPIXAR

Ⅰ 基本方針

1. ガイドラインの目的

1) クオリティの向上
最適だと思われるルールをガイドラインとして制定することで、一定以上のクオリティの向上を担保する

2) クオリティの均一化
制作・運営に複数人が関わる場合の制作物のバラつきを抑制する

3) 制作の効率化
ルールがあることでタイムロスを防ぐ
ルールを決めておくことで、無駄なチェック・修正の工数を削減する

上記目的を達成するために、常に見直し・更新を行ない、積極的にガイドラインの制定と見直し共有を図る。

2. コーディング 6 大原則

1. Web サイト制作者は本ガイドラインをコーディングの基準とし、順守すること。
2. サイトの内容や構成を考慮した上で、適切な文書構造でマークアップを行う。
3. 適切かつ必要最小限の記述でコーディングを行う。
4. 運用/更新が想定される箇所においては、内容の増減に対応できるよう配慮する。
5. サイトの構成要素は、運用のしやすさと複数人での制作を考慮した作りにする。
6. 最新の技術を常に取り込み、かつユーザーの閲覧環境を考慮しする。

Ⅱ HTML 制作ルール

1. 文字コード
原則 UTF-8 とする。
※システム要件/言語/サーバによるリクエストなどから Shift-JIS や EUC-JP などとする場合もある。

2. 改行コード
原則 CR+LF（Windows）とする。

3. DOCTYPE
基本的に XHTML5 で制作を行う。
要件に応じて、別の形式（XHTML 1.0 Strict、HTML 4.01 Strict、HTML 4.01 Transitional）を採用する。
なお PHP ファイルの場合、XML 宣言は PHP で行う。

例）

■HTML5 定義テンプレート

```
<!DOCTYPE html>
<html lang="ja">
<head>
<meta charset=utf-8" />
<meta name="viewport" content="width=device-width,initial-scale=1">
<title>PAGETITLE｜SITENAME</title>
<meta name="keywords" content="" />
<meta name="description" content="" />
<link rel=”stylesheet” href=”style.css”>

</head>
<body>
中略

<script src=”jquery.js”></script>
</body>
</html>
```

■XHTML 定義テンプレート

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
  "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="ja" lang="ja">
<head>
<meta http-equiv="Content-Type" content="text/html; charset=utf-8" />
<meta http-equiv="Content-Style-Type" content="text/css" />
<meta http-equiv="Content-Script-Type" content="text/javascript" />
<title>PAGETITLE｜SITENAME</title>
<meta name="keywords" content="" />
<meta name="description" content="" />
</head>
<body>
中略
</body>
</html>
```

■PHP ファイルの場合の XML 宣言

```
<?php echo '<?xml version="1.0" encoding="UTF-8"?>'."¥n" ?>
```

4. マークアップ全般
   ■文書内容に適したマークアップを行う。
   ■W3C が勧告する XHTML1.0、HTML5 の仕様に準拠したマークアップを行う。

5. 改行とインデント
   ■ソースの改行は、ブロック要素の終了タグの後等、適宜行う。
   ■body 内の要素には、構造に合わせて適切にインデントを施し、視認性の高いソースコードを実現する。
   ■インデントには tab（半角スペース 4 つ分）を使用し、全角・半角スペースを使用しない。

6. パスの記述ルール
   6.1. 一般ルール
   ■サイト内リンク/画像のリンク共に、指定がない場合は原則として相対パス（ドキュメント）とする。

   ■SSI や CMS の都合上、ルート相対パス（サイトルート）で記述する必要がある場合は、事前にクライアントに確認し了承を得ること。

   ※ルート相対パスの場合は、XAMPP や MAMP など Apache を使用して制作・確認を行う

   例）
   ■相対パス（ドキュメント）

```
<a href="../company/index.html">企業情報</a>
```

   ■ルート相対パス（サイトルート）

```
<a href="/company/index.html">企業情報</a>
```

   ■絶対パス

```
<a href="http://coding-factory.com/company/index.html">企業情報</a>
```

6.2.インクルードファイルの場合

■SSI

インクルードファイルは「.php」で作成し、「/common/inc/」内に下記の名前で保存する。

/common /inc/header.html
footer.html
gnavi.html
lnavi.html

例）

```
<!--#include virtual="/common/inc/header.html"-->
```

■PHP

・相対パス（ドキュメント）

<?php include_once(dirname(__FILE__)."/相対パス"); ?>

例）

```
<?php include_once(dirname(__FILE__)."/../header.html"); ?>
```

・ルート相対パス（サイトルート）

<?php include($_SERVER['DOCUMENT_ROOT'].'ルート相対パス'); ?>

例）

```
<?php include($_SERVER['DOCUMENT_ROOT'].'/common/inc/header.html'); ?>
```

7. タグの記述ルール

7.1. 主要なタグの共通ルール

■<title>タグ

・ページ名、親カテゴリ名、サイト名の順に、「｜」（全角縦棒）で区切る。

親カテゴリがない場合　　：ページ名｜サイト名

親カテゴリにあたるカテゴリが多い場合 ：ページ名｜カテゴリ名｜サイト名

■<meta>タグ

・keywords は、キーワード毎に半角カンマ「,」で区切ること。

※半角カンマ以外で区切ると、区切りとならずに最初から最後までがひとつのキーワードとなってしまう為。

■<hX>タグ
・<h1>は、特に指定がなければ、TOP ページではサイトタイトル・ロゴなどに使用し、その他のページではページ固有のタイトルにあたるテキスト、もしくは img 要素に対して使用する。
・ページ内では文書の論理構造に合わせて、必ず見出し要素を使用すること。

■<img>タグ
・img タグには必ず画像サイズ属性と属性値を指定すること。
※画像のサイズを指定することによって、ページを読み込む早い段階でレイアウトが決定されるため、全体のレンダリングが早く終了する。

・意味のある画像は、原則背景ではなく img 要素として配置する。
※文書を補完する画像、単独で内容を説明している画像、メインイメージなど

・alt 属性にて画像の代替となるテキストを設定する。
※特に指定が無い場合は、その画像の簡易説明を alt の内容とする。
※装飾用の画像など、相応しい代替テキストが存在しない画像の場合には、「alt=""」と設定する。

■ タグ
・ソース内容が把握しやすいよう、class、id 付きの閉じ の前に必ずコメントを入れる。
※コメントを挿入することで、ソースの視認性を上げ人的ミスの発生を抑えることを目的とする。
例）

```
</div> <!-- /contents -->
```

7.2. フォーム関連タグの記述ルール
■<input>タグ、<textarea>タグ、<select>タグの id 属性に対してスタイルを適用しない。
※id 属性は変更されることを考慮し、class 属性を使用しスタイルを適用する

■チェックボックスとラジオボタンには<label for="任意値">と対応した id 属性を必須とする。
"label" + ["_"] or [“-”] + [連番]
例）

```
<label for="label_01"><input type="checkbox" id="label_01" />ラベル</label>
```

■ラジオボタンには name 属性を必須とする。
"radio" + ["_"] or [“-”] + [連番]
例）

```
<label for="label_01"><input type="radio" name="radio_01" id="label_01" />ファイン</label>
<label for="label_02"><input type="radio" name="radio_01" id="label_02" />ピクサー</label>
```

■<input>タグの checked と、<select>タグの selected 指定はデザイン（ワイヤー）に合わせる。

■フォーム内の画像ボタン（戻る／確認／送信ボタンなど）は type="image"で設定する。
例）
```
<input type="image" src="画像パス" />
```

■フォーム全体を form タグで囲み、属性に method="post" と action="（遷移先 URL）" を指定する。
例）
```
<form method="post" action="遷移先 URL">
　　～～　フォーム　～～
</form>
```

8. その他の記述ルール
■英数字は半角で統一とする。
例）
誤：２０１１年１０月３１日
正：2011 年 10 月 31 日

■要素名および属性名はすべて小文字で記述する。
例）
要素：div、p、input、img
属性：id、class、type、src

■タグ内での属性の記述順は class="オリジナル 汎用"とする。
例）
```
<div class="sectionContact mt10">
```

■汎用クラスは多用しない。

■終了タグは省略しない。
例）
</p>、 、</tr>、、</option>

■CSS での再現が困難な場合を除き、必然性のない（レイアウトを目的とした）table コーディングは行わない。

■Google Analytics(グーグル・アナリティクス)タグは、原則として<head>(開始ヘッド)の直後に記述する。

■パンくずの表記については階層構造に従い、各ページまでの遷移を順に記述する。
例）
ホーム ＞ 大カテゴリ ＞ 小カテゴリ ＞ ページ名
ホーム ＞ ページ名

■半角カタカナは原則として使用しない。

■メールアドレスはエンティティ文字に変換し記述する。

■機種依存文字の記述には、下記のようなエンティティ文字を使用する。
主なエンティティ文字の例）

記号	意味	エンティティ文字（文字参照）
<	小なり不等号	<
>	大なり不等号	>
&	アンパサンド	&
	半角スペース(改行禁止)	
®	登録商標	®
©	著作権	©
™	トレードマーク	™
￥	円	¥

Ⅲ CSS 制作ルール

1. 文字コード

HTML に合わせる。

2. フォントファミリー

フォントファミリーは指定がない場合、以下の順序で記述する。

font-family: "游ゴシック体", "Yu Gothic", YuGothic, "ヒラギノ角ゴ Pro", "Hiragino Kaku Gothic Pro", "メイリオ", Meiryo, "MS P ゴシック", "MS PGothic", sans-serif;

3. CSS 記述の統一

3.1. グローバルナビゲーションについて

■グローバルナビゲーションは<ul>で組み、<nav>で囲う。
■<ul>タグには id を付与し、その id 名は"gNavi"等とする。

3.2. ローカルナビゲーションについて

■ローカルナビゲーションは<ul>で組む。
■<ul>タグには id を付与し、その id 名は"lNavi"等とする。

3.3. id 名

id 名は以下のルールに沿って命名する。
■文書構造上、意味のある名前にすること。
■ローワーキャメル方式、スネーク方式、ケバブ形式のいずれかで命名し、サイト内で統一すること。
例）
・ローワーキャメル
wrap
header
headerIn
headerUtility
GNavi
・スネーク
header_in
header_utility
g_navi
・ケバブ
header-in
header-utility
g-navi

3.4. class 名

class 名は原則、以下のルールに沿って命名する。
■文書構造上、意味のある名前にすること。
■ローワーキャメル方式、スネーク方式、ケバブ形式のいずれかで命名し、サイト内で統一すること。
■必要に応じて、末尾に 2 桁の連番を付ける。
例）
categoryList01
sec_tutorial01
btn-tweet01

■[識別名]と[詳細]は「_(アンダースコア)」で連結すること。
例）
ul_arrow01, ul_circle01, ul_square01
h2_basic01, h2_basic01_in, h2_basic02, h2_basic02_in
table_spec01, table_spec02

■継承必須 class の命名
他の class からセレクタの継承をしている場合に限り、1 単語の class 名を付けてよい。
例）

```
ul_entry01 {}
ul_entry01 li {}
ul_entry01 li img {}
ul_entry01 li .time {}
        ・
        ・
        ・
```

■特殊 class 名について
以下の用途については class 名を統一する。
・アクティブ時の class 名："current" or “active”
・オーバー時の class 名："over" or “hover”
・div, section 要素の入れ子
基準となる要素の親要素："sectionXXXWrap"
基準となる要素の子要素："sectionXXXIn"

3.5. 一般 css 記述
■セレクタについて
・コードヒントを利用するため、プロパティの値の前には、必ず半角スペースを入れる。
・継承させる場合はセレクタにインデントを付ける。
・連続して同種のセレクタに同じようなプロパティをかける場合は、1 行でまとめてもよい。
例）

```
.example {
      display: block;
      margin: 0 auto:
      width: 900px;
}
.example01 a { margin-left: -100px;}
.example02 a { margin-left: -200px;}
.example03 a { margin-left: -300px;}
```

■ショートハンドプロパティについて
・一括指定可能なプロパティ(margin 等)の個別指定(margin-top 等)は、1 つまでとする。
・2 つ以上の個別指定は一括指定にまとめる。

誤）
```
margin-top: 4px;
margin-right: 10px;
margin-bottom: 8px;
margin-left: 20px;

background-image: url(…);
background-repeat: no-repeat;
background-position: top right;
background-size: cover;
```

正）
```
margin: 4px 10px 8px 20px;
background: url(…) no-repeat top right / cover;
```

■色の指定
色指定は 16 進数 6 桁で記述するが、省略できるものは 3 桁に省略する。
※英字は小文字を使用する
例）
#fde96c #d6e0f4
#000 #999 #f00 #fff 等

■コメントアウト
コメントアウトは以下のように記述する。

・大見出し（ディレクトリ・ページ名など）
```
/* ----------------------------------------------------------
   Comment
---------------------------------------------------------- */
```

・中見出し
```
/* Comment
---------------------------------------------------------- */
```

・小見出し、注記
```
/* comment */
```

■CSS hack について
CSS hack はできるだけ使用を避けるものとし、やむを得ず使用する場合は
IE に対して hack を使用すること。

Firefox や Google Chrome、Safari でレイアウトが崩れていた場合、通常の記述でそれらの表示に合わせ
IE 用に hack の記述を行うこと。

IE8 記述例）

```
html>/**/body { font-weight /*¥**/: bold¥9;}
```

4. 初期設定 CSS
4.1. import.css
複数の CSS を一括で呼び出すための CSS。
基本的に設定の必要は無いが、共通 CSS などが増えた場合は都度編集する。

下記は初期の記述

```
@charset "utf-8";
/*
 * import.css
 *
 *    version --- n.n
 *    updated --- YYYY/MM/DD
 */

@import "cmn_layout.css";
@import "cmn_style.css";
```

4.1. style.css
基礎構造を成す id,class を設定した css
各ブラウザのリセットは下記の記述のみで不必要。
必要に応じて、適宜修正は可。

下記は初期の記述

```
/*  ブラウザの互換初期化----------------------- */
html{ overflow:scroll; -webkit-text-size-adjust: 100%; }
* {margin: 0; padding: 0; box-sizing: border-box;}
body,div,dl,dt,dd,h1,h2,h3,h4,h5,h6,pre,form,fieldset,input,p,blockquote,th,td{margin:0;padding:0;}
table{border-collapse:collapse;border-spacing:0;}
fieldset,img{border:0;}
address,caption,cite,code,dfn,em,th,var{font-style:normal;font-weight:normal;}
caption,th {text-align:left;}
h1,h2,h3,h4,h5,h6{font-size:100%;}
q:before,q:after{content:'';}

input[type="checkbox"],input[type="radio"] {margin: 0 2px 0 0; padding: 0; vertical-align: -2px; }

/* clearfix------------ */
.clearfix:after { content: "."; display: block; clear: both; height: 0; visibility: hidden; }
.clearfix { min-height: 1px; }
* html .clearfix { height: 1px; /*¥*//*/ height: auto; overflow: hidden; /**/ }

/* ios でのフォーム系デフォルトスタイルをリセット  */
button,
input[type=”button”],
input[type=”text”],
input[type=”submit”]{
-webkit-box-sizing: content-box;
-webkit-appearance: button;
border:0 none;
box-sizing: border-box;
cursor: pointer;
-webkit-border-radius: 0;
background: transparent;
}

button::-webkit-search-decoration,
input[type=”button”].clearbtn{
border: 0 none;
display: inline;
}
```

4.3. common.css
基礎構造(コンテンツやヘッダー、フッター)などページのレイアウトのプロパティと、
テンプレートなど頻出する id,class を定義した css。

【基礎構造 セレクタ】

セレクタ例	説明
header	ヘッダー
gNavi	グローバルナビ
breadcrumb	パンくず
mainimg	メインイメージ
contents	メインコンテンツ
pageTop	ページトップへボタン
footer	フッター

【テンプレートなど汎用セレクタ例】

クラス名	説明
fzX	font-size の設定
taX	text-align の設定
vaX	vertical-align の設定
blockX	block 要素の中央・右配置
wNNN	px 単位での幅の設定
wHalf, wTri, wQuart, wFull, wMax	%単位での幅の設定。基本的にカラム分けするもの
wNNper	%単位での幅の設定。3〜50%できりのいい数値
flL(R)	float 指定（IE6 のマージンバグに対応する為、display: inline 付き）
clear, block, hide, bgN, tdU, tdN	特定の CSS のみの設定
over, png, js_popup_width_height,, js_window_close	JavaScript 呼び出し用 class
atode	後で制作するパーツに付ける目印の設定。
mx00	マージンの設定
px00	パディングの設定

5.1 css 拡張言語について
社内では css 拡張言語を利用する場合は基本的に sass を標準とし、拡張子は.scss とする。

```
├_setting.scss（基本仕様定義 SCSS）
|
├□□.scss
└□□.scss
```

【sass 基本変数一覧】

変数名	説明
$sansselif	基本のサンセリフ体 font-family
$selif	基本のセリフ体 font-family
$baseFont	標準フォントサイズ (px)
$lineHeight	標準 line-height (em)
$spWidth	レスポンシブ スマートフォン切り替え定義 例：only screen and (max-width: 767px;)
$pcWidth	レスポンシブ PC 切り替え定義 例：only screen and (min-width: 767px;)
$baseWidth	基本サイト幅(px)
$contentWidth	本文コンテンツ幅(px)
$baseColorX	基本カラースキーム（X は 2 桁数字 必要に応じ増やす）
$grayBorder	基本のグレーのボーダーカラー
$basefontColor	ベースフォントカラー
$linkfontColor	リンクフォントカラー

※上記はあくまで参考です。必要に応じ変更・追加・削除しても良い。

5.2 scss ファイルの管理
原則として sass でコンパイルされた css を web サーバーにアップロードする際は、同時に scss ファイルもアップロードする事。
共有サーバーへのバックアップに関しても同様です。

IV JavaScript 制作ルール

0. 記述場所の前提

原則、複数ページで共通に使用するスクリプトは外部ファイル化する。

1. 文字コード

HTML に合わせる。
ただし、ダウンロードしてくるものは原則変更しない。
※HTML ファイルと JavaScript ファイルの文字コードが異なる場合、キャラクターセットの指定を行うこと。

```
<script type="text/javascript" src="/common/js/common.js" charset="UTF-8"></script>
```

2. 初期設定 JavaScript

2.1. jQuery.js

JavaScript ライブラリ
http://jquery.com/
※使用時は毎回ダウンロードすること。

3. 追加設定 JavaScript

必要に応じて JavaScript を追加する。

※使用時は極力最新版を毎回ダウンロードし利用すること。

V ディレクトリ作成ルール

1. 基本的なディレクトリ構成

1.1 基本的なディレクトリ構成

```
root/
 ├index.html
 ├style.css (リセット css、基礎構造とテンプレートなど頻出する id,class を定義した css)
 ├sub.css (TOP ページ、ディレクトリ毎の CSS をまとめて)
 │
 ├img/
 │  ├□□/ (必要に応じて追加)
 │  └画像ファイル (サイト全体で利用、頻出する画像)
 │
 ├js/
 │  ├jquery.js
 │  └(○○/) (JS パーツ格納ディレクトリ ※必要に応じて)
 │     ├○○.js (その JS に必要な JS ただし jquery.js サイトに付 1 つ重複させないようにする)
 │     ├○○.css (その JS に必要な CSS)
 │     └(images/) (その JS に必要な画像※必要に応じて)
 │
 └□□/
    ├□□.css（ディレクトリ毎の CSS）
    ├img/（ディレクトリ毎の画像）
    └index.html
```

1.2 WordPress 基礎テンプレート制作時のディレクトリ構成

WordPress を目的とした基礎コーディングの場合、テーマファイルへの速やかな置き換えを前提とした構造を目的としています。

```
root/
 ├index.html
 │
 ├style.css (テンプレート定義とリセット CSS の記述)
 ├common.css (基礎構造とテンプレートなど頻出する id,class を定義した css)
 ├○○.css (TOP ページ、ディレクトリ毎の CSS をまとめて)
 │
 ├img/（サイト全体で利用・頻出する画像）
 │    ├(△△△/)（ディレクトリ内で利用する画像 ※必要に応じて）
 │    └(□□□/)（ディレクトリ内で利用する画像 ※必要に応じて）
 │
 ├js/ (必要に応じて追加)
 │  ├jquery.js
 │  └(○○/) (JS パーツ格納ディレクトリ ※必要に応じて)
 │    ├○○.js (その JS に必要な JS ただし jquery.js サイトに付 1 つ重複させないようにする)
 │    ├○○.css (その JS に必要な CSS)
 │    └(images/) (その JS に利用する画像)
 │
 ├△△△.html
 ├(△△△/)（ディレクトリ内で利用する画像 ※必要に応じて）
 ├□□□.html
 ├(□□□/)（ディレクトリ内で利用する画像 ※必要に応じて）
```

※基本デレクトリ構成と違い、style.css ではテンプレート定義とリセット CSS のみ記述。基礎構造とテンプレートなど頻出する id,class を定義した css は common.css に記載します。

2. ディレクトリ作成権限について
サイト作成開始時にディレクトリリストを必ず作成し、それに沿って運用を行う。
作業中にディレクトリの増減、リネーム等を行う場合は、担当者間で取り決めること。

3. ディレクトリ名使用文字
ディレクトリ名に使用できる文字は以下のとおり。
■「a」～「z」までの小文字のアルファベット（1 バイト）
■「0」～「9」までの英数字（1 バイト）
■「-（ハイフン）」と「_（アンダースコア）」（いずれも 1 バイト）
■先頭には「-（ハイフン）」「_（アンダースコア）」を使用しない。
■スペースは使用しない。
■単語が複数連続する場合、単語間に「_（アンダースコア）」を使用する。

4. ディレクトリ文字数
ディレクトリ名は半角 31 文字以内で収めるようにすること。

5. ディレクトリルール
5.1. 共通画像とディレクトリ別画像のディレクトリは分ける。
共有画像　　：common/img/
ディレクトリＡのみで使用する画像：ディレクトリ名Ａ/img/
ディレクトリＢのみで使用する画像：ディレクトリ名Ｂ/img/

5.2. 基本的に英単語表記にし、なるべく内容が分かるような命名をすること。
誤）「sv001/」「usrsup/」「contfrms/」
正）「services/」「support/」「contact/」

5.3. 数字を用いる必要がある場合は、01,02...09,10,11 など、連番 2 桁にすること。

5.4. 一つのカテゴリ配下に画像フォルダを一つ作成し画像をまとめるか、それぞれのフォルダ毎に画像をまとめるかは、状況に応じて制作者が決めること。

VI ファイル作成ルール

1. ファイル名使用文字
1.1. ファイル名に使用できる文字は以下のとおり。（ファイル名には拡張子必須）
■「a」～「z」までの小文字のアルファベット（1 バイト）
■「0」～「9」までの英数字（1 バイト）
■「-（ハイフン）」と「_（アンダースコア）」（いずれも 1 バイト）
■先頭には「-（ハイフン）」「_（アンダースコア）」を使用しない。
※ファイル名に 2 バイト文字とスペースは不可

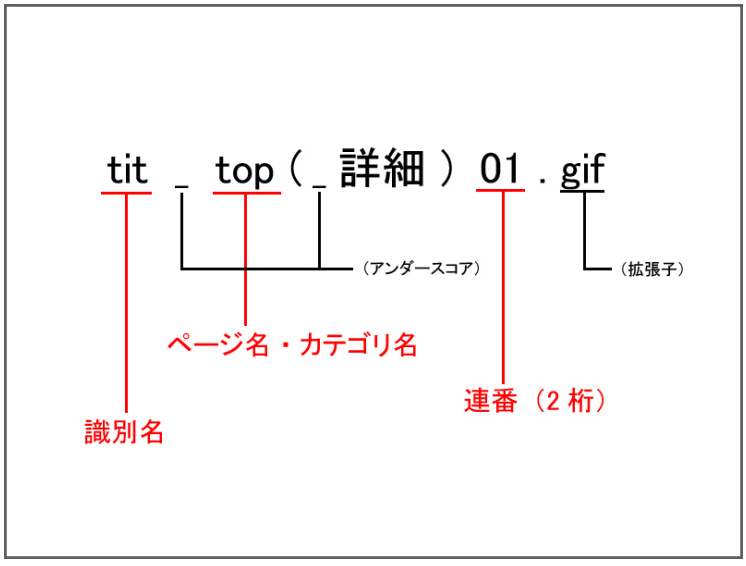
2. ファイル名文字数
■拡張子を含む半角小文字 31 文字以内とする。
■ページ名が長い場合は省略できるものとする。

3. 画像ファイル作成ルール
3.1. 画像の命名ルール
画像名は以下の通りとする。

[識別名] + ["_"] ＋ [ファイル・カテゴリ名]（+ ["_"]+ [詳細] ）+ [連番 2 桁].拡張子
例） tit_top01.gif
bg_lineup_form01.gif

もしくは

[識別名] + ["_"] ＋ [ファイル・カテゴリ名]（+ [詳細（ローワーキャメル）] ）+ [連番 2 桁].拡張子
例） tit_top01.gif
bg_lineupForm01.gif



※[ファイル・カテゴリ名]、[詳細]は不要だと判断される場合は省略可。
※同種の画像が他にない場合でも、必ず 2 桁の連番をつける。

【識別名一覧（サイト内に制作する主な画像の命名ルールは下記を参照）】

識別名	用途	命名サンプル
bg	背景や、ボーダー用にリピートさせて使われる画像	bg_top01.gif bg_dotx01.gif bg_dotx_3x1_01.gif
btn	ボタンとして使われる画像	btn_contact01_no.gif btn_contact01_on.gif
ico	アイコンとして使われる画像	ico_arrow01.gif ico_square01.gif
bnr	バナーとして使われる画像	bnr_sub01.jpg
h1～h6	<h1>タグなどの論理タグで囲まれる見出し画像	h2_business01.gif h3_service01.gif
tit	h1～h6 等に用途が限られない見出し画像	tit_topic01.gif
txt	デザイン上、画像化するテキストの画像	txt_corporate01.gif
pic	商品や人物、風景などの写真画像	pic_about01.jpg
thumb	写真と連動するサムネイル（小さい）画像。	thumb_about01.jpg thumb_about02.jpg
img	写真以外のイラストや図表などの画像	img_about01.jpg
hero	ページのメイン画像	hero_about01.jpg

・グローバルナビゲーションの画像（必ず common/img/に入れる）
gnavi.拡張子

・ローカルナビゲーションボタン（必ず common/img/に入れる）
通常時　　：lnavi + [連番 2 桁] + ["_"] + [no].拡張子
マウスオーバー時　　：lnavi + [連番 2 桁] + ["_"] + [on].拡張子
アクティブ時　：lnavi + [連番 2 桁] + ["_"] + [cr].拡張子
※マウスオーバー時と同じならば不要
※横並びリストの際は gnavi に準じる

・組み合わせて使用する画像
角丸 BOX での背景など、複数画像を組み合わせて一つの画像を形成する場合は、連番の後に詳細説明が入る。
例）
faq.html のフォームの背景画像（角丸）の下側
faq_bg_form01_bottom.gif

3.2. 画像の書き出しルール
■gif
主にアイコンや簡単なテキスト画像などの色数の少ない、あるいは透明を含む画像などに使用する。

■jpg
色数が多く、サイズの大きい写真などの画像で使用する
書き出す際は、原則として以下のように圧縮するが、クオリティに配慮しながら、極力ファイルサイズは軽減すること。
Fireworks　：　100%
Photoshop　：　72～90%

■png
半透明の処理が必要な場合に使用する。

4. 不要ファイル削除ルール

納品時には下記のゴミファイルを削除する。

■_notes　（Dreamweaver で作らない設定にする）

■.bak　（WinMerge で作らない設定にする）

■.DS_Store および 「._(ドットアンダースコア)」から始まるファイル名のデータ

Ⅶ コーディング後チェック

1. 文法チェック
HTML-Lint（文法チェックツール）を用いて検証テストを行う。

http://www.htmllint.net/（html5）
https://cetus.sakura.ne.jp/htmllint/(XHTML)

※6 点以上のエラーを無くす。

2. ブラウザ目視チェック
表示のズレ、リンク先のチェック等、目視でのチェックが必要な項目をブラウザごとに検証する。

対象ブラウザ
■ PC ・Internet Explorer 11～（windows） ・Microsoft Edge 最新（windows） ・Firefox 最新（windows、mac） ・Google Chrome 最新（windows、mac） ・Safari 最新（mac） ■ スマートフォン ・Safari 最新（iphone） ・Google Chrome 最新（android） ■ タブレット ・Safari 最新 ・Google Chrome 最新

3. Analytics

基本的にはお客様の Google アカウントから設置を行う。
お客様の Google アカウントがない場合や解析業務を委託されている場合は
社内アカウントで管理を行う。

4. その他チェック
共有内「公開前チェックリスト」を参考に確認する。