# 第 11 章

# Bootstrap と Font Awesome

本章では、Bootstrap と Font Awesome を導入することにより、ModestGreeter の外観を整えます。

## 11.1 Bootstrap の導入

Phoenix サーバーが起動中の場合は、停止してから以下の作業を行ってください。

### Bootstrap のインストール

CSS/JavaScript ライブラリ **Bootstrap** を用いると、パソコン、スマートフォン、タブレット PC など画面サイズの異なる端末に対応した Web ページを効率的に制作できるようになります。ModestGreeter に Bootstrap を導入しましょう。

本書では Bootstrap のバージョン 4（Bootstrap 4）を採用します。本書の最終更新時点（2017 年 11 月）では Bootstrap4 はまだ正式にリリースされていません。今後、仕様が変更される可能性があります。

ターミナルで、以下のコマンドを順に実行してください。

```
$ cd ~/projects/modest_greeter
$ npm install --save jquery popper.js tether bootstrap@4.0.0-beta.2
```

--save は、インストールしたパッケージの名前とバージョン番号を package.json というファイルの dependencies セクションに書き込むためのオプションです。npm install コマンドを実行したとき、このセクションに列挙されているパッケージは開発環境だけでなく本番環境でもインストールされます。

### スタイルシートの設定

web/static/css ディレクトリの app.css を削除します。

```
$ rm web/static/css/app.css
```

同ディレクトリに新規ファイル app.scss（拡張子に注意）を作成し、次の内容を書き込みます。

web/static/css/app.scss (New)

```
1 @import "node_modules/bootstrap/dist/css/bootstrap";
```

### JavaScript の設定

web/static/js ディレクトリの app.js を次のように書き換えます。

web/static/js/app.js

```
 :
10 // Import dependencies
11 //
12 // If you no longer want to use a dependency, remember
13 // to also remove its path from "config.paths.watched".
14 import "phoenix_html"
```

```
15 + import "bootstrap"
16
17   // Import local files
 :
```

## Brunch の設定

ModestGreeter のルートディレクトリにある brunch-config.js を次のように書き換えます。

brunch-config.js

```
 :
69     npm: {
70 -     enabled: true
70 +     enabled: true,
71 +     globals: {
72 +       $: "jquery",
73 +       jQuery: "jquery",
74 +       Popper: "popper.js",
75 +       Tether: "tether"
76 +     }
77
78     watcher: {
79       usePolling: true
80     }
81   };
```

Phoenix は、フロントエンドビルドツール Brunch を利用して、JavaScript プログラム、スタイルシート、画像ファイルなどの変換、結合、圧縮、配置などの作業を行います。brunch-config.js はフロントエンドビルドツール Brunch の設定ファイルです。

Brunch は NPM とうまく統合されているので、基本的に Phoenix 開発者は npm install コマンドで必要な NPM パッケージをインストールし、web/static/js/app.js でインポートするだけで十分です。しかし、Bootstrap のように brunch-config.js で追加の設定を書かなければならない場合もあります。

■コラム: npm.globals セクション

Bootstrap は、別の JavaScript ライブラリ **jQuery** に依存しており、その提供するオブジェクトが $ および jQuery というグローバル変数として定義されていないとうまく動きません。また、Bootstrap のいくつかの機能は、さらに **popper.js** および **tether** という JavaScript ライブラリに依存していて、その提供するオブジェクトが Popper および Tether というグローバル変数として定義されていることを要求します。

このような場合、brunch-config.js の npm.globals セクションに、グローバル変数名とパッケージ名の組み合わせを列挙してください。

ちなみに、web/static/js/app.js で "jquery" と "popper.js" と "tether" をインポートしても、これらのパッケージが提供するオブジェクトがグローバル変数にセットされることはありません。

## 11.2 Card コンポーネント

Web デザイナーたちが Bootstrap を採用する第一の目的は、レスポンシブ Web デザインを実現することです。すなわち、ブラウザの画面サイズに応じてレイアウトを最適化することです。しかし、Bootstrap が用意する多彩なコンポーネント（**component**）を使うという目的もあります。

本章では、そのひとつ **Card コンポーネント**を紹介します。

### 基本的な構成

web/templates/hello ディレクトリの show.html.eex を次のように変更してください。

web/templates/hello/show.html.eex

```
  :
- <p>Hello, <%= @name %>!</p>
+ <div class="card">
+   <div class="card-body">
+     <p>Hello, <%= @name %>!</p>
```

```
4 +   </div>
5 + </div>
```

変更点は以下のとおりです：

1. p 要素を div 要素で二重に囲んだ。
2. 外側の div 要素の class 属性に card を指定した。
3. 内側の div 要素の class 属性に card-body を指定した。

div 要素は、いわば「透明な箱」のような存在です。「段落」という意味を持つ p 要素と異なり、特別な意味を持ちません。主に、スタイルの適用範囲を区切るために用いられます。

class 属性を用いると、HTML の要素に**クラス**を設定できます。クラスとはスタイルに名前をつけたものと考えてください。ただし、クラスには他の役割もあります。

Bootstrap は数多くのクラスを定義しています。例えば、card クラスは「四辺を薄い灰色の角丸枠線で囲む」というスタイルを持ちます。また、card-body クラスには「四辺のパディングの幅を 1.25rem にする」というスタイルが設定されています。

Phoenix サーバーを起動して、ブラウザをリロードすると、図 11.1 のような画面に切り替わります。

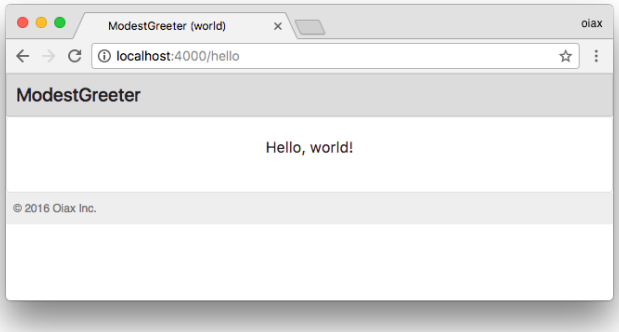


図 11.1　Card コンポーネント（初期状態）

このように Bootstrap を導入すると、スタイルシートを書かなくても HTML 文書の外観を変えることができます。

### card-text クラス

さて、図 11.1 を注意深く観察すると、「Hello, world!」というメッセージの上下のマージンが均等でないことに気付きます。しかし、show.html.eex を次のように変更すればこの問題が解消されます。

web/templates/hello/show.html.eex

```
:
5     <div class="card-body">
6 -     <p>Hello, <%= @name %>!</p>
6 +     <p class="card-text">Hello, <%= @name %>!</p>
7     </div>
```

card-text クラスは面白い効果を持ちます。これは基本的には何のスタイルも設定しません。ただし、親要素の最後の子要素である場合は、下辺のマージンが 0 に設定されます。

Bootstrap はデフォルトで p 要素の下辺のマージンを 1rem に設定しています。複数の段落が重なった時に段落と段落の間を少し空けるためです。しかし、最後の段落ではこの 1rem のマージンが邪魔になります。

この変更の結果、ブラウザの画面は 図 11.2 のようになります。

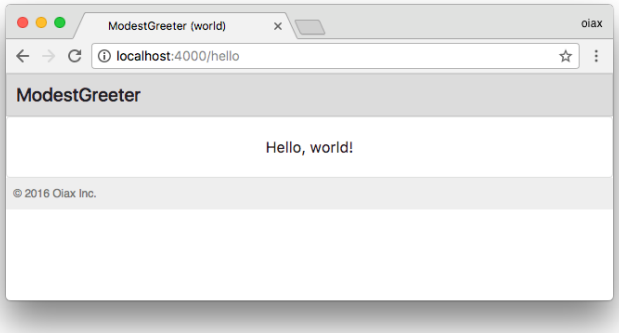


図 11.2　Card コンポーネント（card-text クラスを適用）

## マージンの調整

現状の Card コンポーネントは上下左右のマージンがないため「カード」のように見えません。そこで、show.html.eex を次のように変更します。

web/templates/hello/show.html.eex

```
  :
1 - <div class="card">
1 + <div class="card m-3">
  :
```

class 属性の値を空白文字で区切れば、ひとつの要素に複数のクラスを与えることができます。外側の div 要素に対して m-3 というクラスを追加しました。

さて、m-3 のような 2 個の英数字をハイフンで連結したクラスは、間隔（spacing）の設定に使われます。

1 文字目の m はマージンを意味します。この文字を p で置き換えると、パディングを設定するクラスになります。

2 文字目の数字は間隔の幅を表します。0 が 0、1 が 0.25rem、2 が 0.5rem、3 が 1rem 、4 が 1.5rem 、5 が 3rem を意味します。

ここでは、Card コンポーネントの四辺の余白を 1rem に揃えています。その結果、ブラウザの表示は 図 11.3 のようになります。

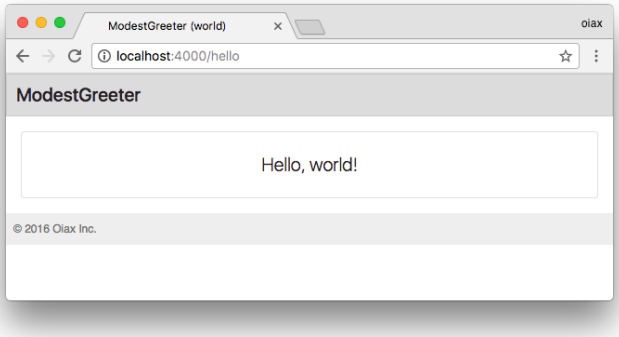


図 11.3　Card コンポーネント（マージンを調整）

なお、上下左右の辺に対して個別にマージンまたはパディングを設定する場合、mt-1 のようにハイフンの前に 1 個のアルファベットを加えます。この文字は対象となる辺を表します（表 11.1）。

表 11.1　間隔設定クラスの 2 文字目の意味

| 文字 | 対象となる辺 |
|---|---|
| t | 上辺 |
| r | 右辺 |
| b | 下辺 |
| l | 左辺 |
| x | 左辺と右辺 |
| y | 上辺と下辺 |

## 11.3　背景色とフォントサイズを調整

さらに外観の調整を続けましょう。show.html.eex を次のように変更してください。

web/templates/hello/show.html.eex

```
  :
```

```
1   <div class="card m-3">
2 -   <div class="card-body">
2 +   <div class="card-body text-white bg-success">
3 -     <p class="card-text">Hello, <%= @name %>!</p>
3 +     <p class="card-text lead">Hello, <%= @name %>!</p>
4     </div>
5   </div>
```

text-white クラスは、内側の要素の文字色を白にします。

bg-success クラスは背景色を緑（#28a745）にします。success の部分を他の名前で置き換えれば、様々な色を背景色として利用できます（表 11.2）。

表 11.2　背景色を設定するクラス

| クラス名 | 背景色 | 16 進数表示 |
|---|---|---|
| bg-primary | 青 | #007bff |
| bg-secondary | 灰色 | #868e96 |
| bg-success | 緑 | #28a745 |
| bg-danger | 赤 | #d9534f |
| bg-warning | オレンジ色 | #ffc107 |
| bg-info | 水色 | #17a2b8 |
| bg-light | 薄い灰色 | #f8f9fa |
| bg-dark | 黒 | #17a2b8 |
| bg-white | 白 | #ffffff |

Bootstrap は“blue”とか“red”などの具体的な名前ではなく“primary”とか“danger”などの抽象的な名前を用いてクラスを定義しています。表 11.2 の 3 列目に書いてあるとおり、これらの色はすべて中間色なので、具体的な色の名前で呼ぶと紛らわしいからです。なお、これらの名前は「アダ名」みたいなものです。人間が覚えやすいように仮に付けられているに過ぎません。英語の“danger”は「危険」を意味しますが、赤い色を危険なものにしか使っていけないわけではありません。

lead はフォントサイズを少し大きめ（1.25rem）にするクラスです。

この変更の結果、ブラウザの画面は 図 11.4 のように変化します。

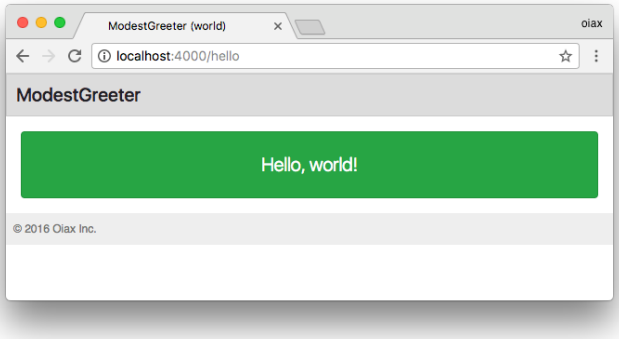


図 11.4　Card コンポーネント（背景色とフォントサイズを調整）

グレースケールのディスプレイやペーパーバック版で本書をお読みの方には分かりませんが、図 11.4 で「Hello, world!」というメッセージは緑色の背景の上に描かれています。

また、Chrome のデベロッパーツール（付録 A 参照）を利用して、ModestGreeter がスマートフォンでどのように表示されるのかを確認してください。

## 11.4　Font Awesome の導入

Phoenix サーバーが起動中の場合は、停止してから以下の作業を行ってください。

### Font Awesome とは

**Font Awesome** を利用すると HTML タグを書くだけで 図 11.5 のようなさまざまなアイコンを Web ページの中で使用することができます。

図 11.5　Font Awesome のアイコンの例

利用可能なアイコンのリストは http://fontawesome.io/icons/ で閲覧できます。

## Font Awesome のインストール

Font Awesome をインストールするには、ターミナルで次のコマンドを実行します。

```
$ npm install --save font-awesome
```

web/static/css ディレクトリの app.scss を書き換えます。

web/static/css/app.scss

```
1   @import "node_modules/bootstrap/dist/css/bootstrap";
2 + @import "node_modules/font-awesome/scss/font-awesome";
```

### フォントファイルの配置

NPM パッケージが提供するフォントファイルや画像ファイルを Brunch で扱うには、copycat-brunch というパッケージが必要となります。

```
$ npm install --save-dev copycat-brunch
```

brunch-config.js を次のように書き換えてください。

brunch-config.js

```
 :
52    // Configure your plugins
```

```
53    plugins: {
54      babel: {
55        // Do not use ES6 compiler in vendor code
56        ignore: [/web\/static\/vendor/]
57 -    }
57 +    },
58 +    copycat: {
59 +      fonts: ["node_modules/font-awesome/fonts"]
60 +    }
61    },
 :
```

plugins.copycat.fonts セクションには、フォントファイルの格納されたディレクトリの配列を指定します。

## Font Awesome の利用

これで Font Awesome が導入されました。では、使ってみましょう。

web/templates/hello ディレクトリの show.html.eex を次のように変更します。

web/templates/hello/show.html.eex

```
 :
3 -    <p class="card-text lead">Hello, <%= @name %>!</p>
3 +    <p class="card-text lead">
4 +      <i class="fa fa-smile-o"></i>
5 +      Hello, <%= @name %>!
6 +    </p>
 :
```

class 属性に fa クラスを指定した i 要素が Font Awesome によってアイコン表示に変えられます。i 要素の内容は空にしてください。

> HTML の i 要素の本来の役割は、「(欧文の印刷において) イタリック体で表記する部分」を示すことですが、Font Awesome はこの要素をアイコン表示のために流用しています。

`fa-`で始まるアイコン固有のクラス名を `class` 属性に指定することにより、表示するアイコンを選択します。ここでは「smile-o」という名前のアイコンを表示したいので `fa-smile-o` というクラス名を指定しました。

Phoenix サーバーを起動しブラウザをリロードすると、図 11.6 のように「Hello, world!」メッセージの前に「笑顔」のアイコンが表示されます。

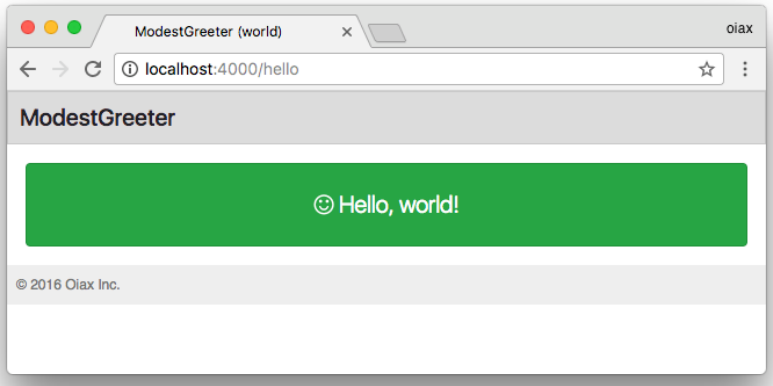


図 11.6　Font Awesome を使ってアイコンを表示