

## 8. CSS フレームワークの基本を学ぶ

### 1. Bootstrap でフォームを作成する

ここでは、Bootstrap のグリッドシステムを使って、レスポンシブ Web デザインにしていきます。今回作成するのは以下の問い合わせフォームです。

ご利用アンケート

HOME へ戻る

この度は、本書をご利用頂きありがとうございます。恐れいりますが、以下のフォームにご記入頂き、送信して頂けると幸いです。

お名前 必須

ご職業

選択してください

知りたい内容（複数回答可）

☐ HTML

☐ CSS

☐ JavaScript

理解度はいかがですか？

☐ 理解できなかった

☐ だいたい理解できた

☐ 理解できた

☒ 回答しない

ご意見

送信する

© H2O space

適当なフォルダに「index.html」ファイルを準備し、同じ階層に「css」フォルダを作成して、中に「style.css」ファイルを準備します。

index.html には次のように記述し、style.css は空のままで構いません (sanitize.css も利用しません)。

```

<!DOCTYPE html>
<html lang="ja">
  <head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width">

    <title>ご利用アンケート</title>

    <link rel="stylesheet" href="css/style.css">

  </head>
  <body>
    <header>
      <h1>ご利用アンケート</h1>
    </header>

    <hr>

    <hr>

    <footer>
      <p>&copy; H20 space</p>
    </footer>
  </body>
</html>

```

## 2. CSS フレームワークを使う

これまでと同じように進めるなら、この後、次のような手順で作成することになります。

- ・リセットまたはノーマライズをする
- ・スタイルを1つずつ定義して、見た目を整えていく
- ・必要に応じてメディアクエリーを記述し、レスポンシブ Web デザインにしていく

しかし、近年これらの手順を「CSS フレームワーク」にまかせてしまうことがよくあります。「フレームワーク (FrameWork)」とは「骨組み」や「枠組み」といった意味で、家を作るときに土地に直接家を建てていくのではなく、既にある骨組みに「外装」だけを作っていくといったイメージです。自分が作りたいものに合う骨組みがあれば、そのまま使った方が、作業が早いというわけです。

CSS フレームワークでは、CSS にあらかじめリセットや「よく使うスタイル」が定義されていて、使う人は各要素の使い方を参照しながら、HTML を作っていくだけで、スタイルが整っていたり、自動的にレスポンシブ Web デザインに対応できたりと、作業を軽減することができるのです。

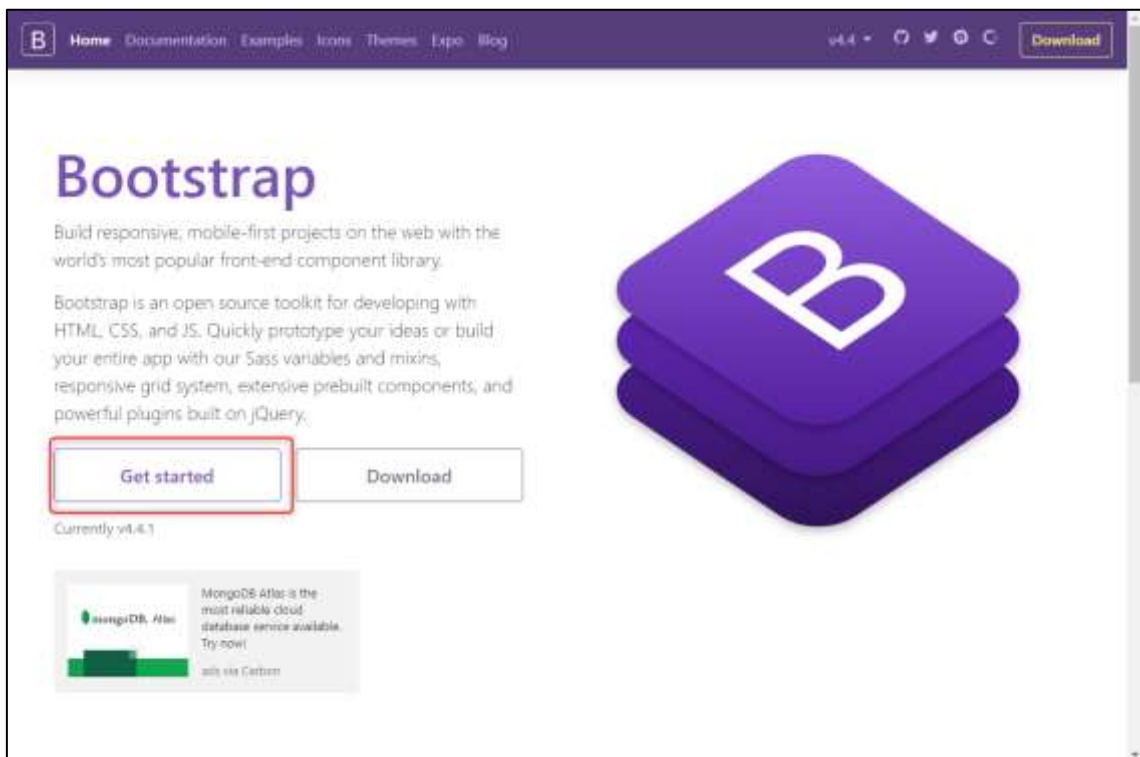
CSS フレームワークには、さまざまな種類がありますが、近年スタンダードとなっているのが、米 Twitter 社が開発した「Bootstrap (ブートストラップ)」というフレームワークです。非常に多機能で使いやすく、多くの Web サイトで採用されています。ここでは、そんな Bootstrap を使ってサイトを作っていきます。

### 3. Bootstrap を導入する

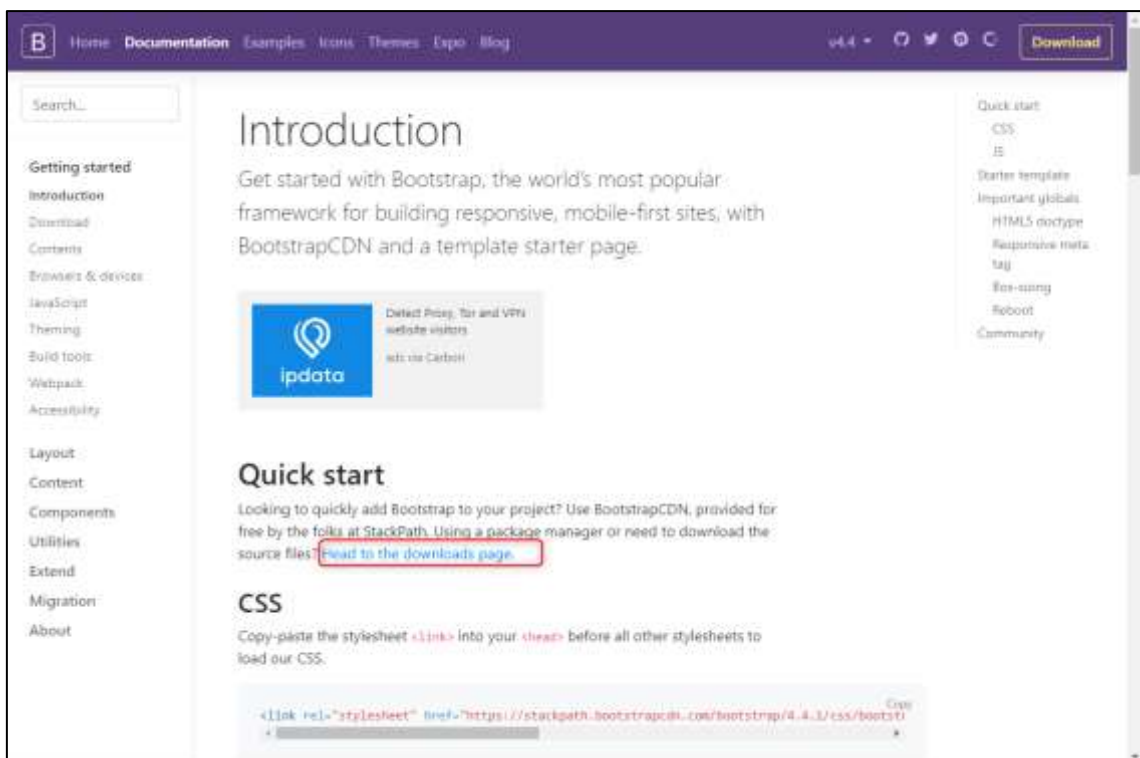
Bootstrap は、次のサイトから利用することができます。

- ・ Bootstrap <http://getbootstrap.com/>

Bootstrap は `sanitize.css` と同じように、ダウンロードしたファイルを参照することでも利用することができます。しかし、ダウンロードしないで「直接」利用する「CDN」という方法も利用できます。ここでは、CDN を利用しましょう。「Bootstrap」のウェブサイトのメニューから「Get started」をクリックします。

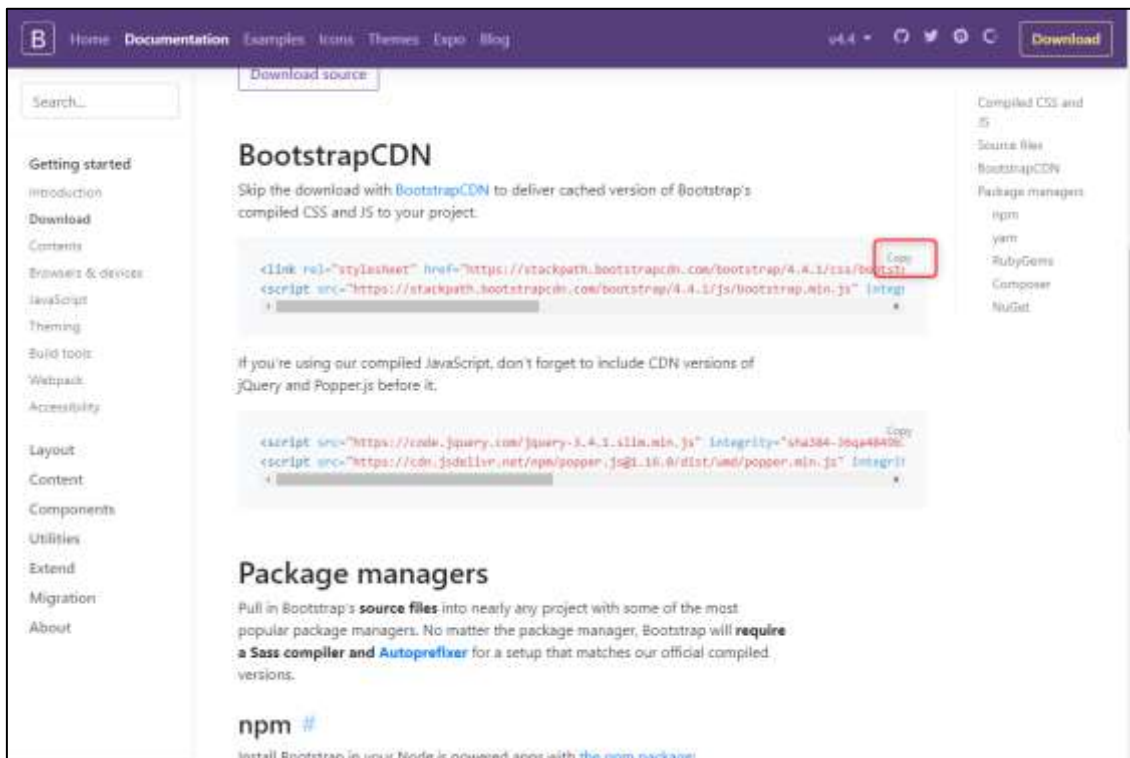


次に、「Head to the downloads page.」をクリックします。



ページを少し進めると、BootstrapCDN と書かれたところがあり、これから作成する

HTML ファイルに記述するコードがありますので、「Copy」を選択しましょう。



コピーができれば、先ほど作った「index.html」に以下のように追加します。



これで、Bootstrap が適用されました。Web ブラウザで表示すると、余白が調整され、ノーマライズがかかったことが分かります。Bootstrap には次の「normalize.css」が同梱されているため、Bootstrap を読み込むだけでノーマライズされるというわけです。

- [necolas/normalize.css](http://github.com/necolas/normalize.css) <http://github.com/necolas/normalize.css>

なお、Bootstrap は必ず自作の CSS (style.css) よりも先に読み込ませておきましょう。この後、自分でスタイルを整える部分が思うように適用されなくなる恐れがあります。

CDN(Contents Delivery Network)とは
CDN は専用の「配信サーバー」を準備することで、CSS や JavaScriptなどをダウンロードさせることなく、直接利用できるようにした仕組みです。指定されたアドレスから直接<link>タグなどでリンクすることで、利用することができます。
使うのが簡単になるのはもちろんですが、Web サイトの利用者にとってもメリットがあります。たとえば、Bootstrap のように人気のある CSS フレームワークは、多くの Web サイトが採用しています。すると、これを使っているすべてのサイトで同じファイルをそれぞれが配信するようになります。利用者は、本当は同じ内容なのに、各サイトからダウンロードしなければなりませんし、「キャッシュ」という仕組みで Web ブラウザが一時的に保存するため、これも無駄になります。
同じ CDN を利用していれば Web ブラウザは「同じファイルである」と認識をします。すると、すでにほかのサイトでキャッシュを蓄えている場合は、同じファイルを再利用しますので、ムダがなくなります。CDN が利用できる環境の場合は、できるだけ利用すると良いでしょう。

#### 4. 画面を中央に集める Containers

完成形のように、まずは画面全体の幅を狭めて、中央に寄せましょう。これを CSS で実現しようとした場合、次のように考えられるかもしれません。

```
width: 600px; /* 幅を 600px にする */
margin: 0 auto; /* 横の余白を auto にして、中央に寄せる */
```

しかし、Bootstrap では CSS を編集しなくても、特定のタグやクラスにスタイルが割り当てられているため、それを使うだけでスタイルが整います。

ここでは、Bootstrap の「Containers」という機能を使ってみます。次のように「Container」という class 属性を持つ<div>要素を追加します。

```

<body>
  <div class="container">
    <header>
      <h1>ご利用アンケート</h1>
    </header>

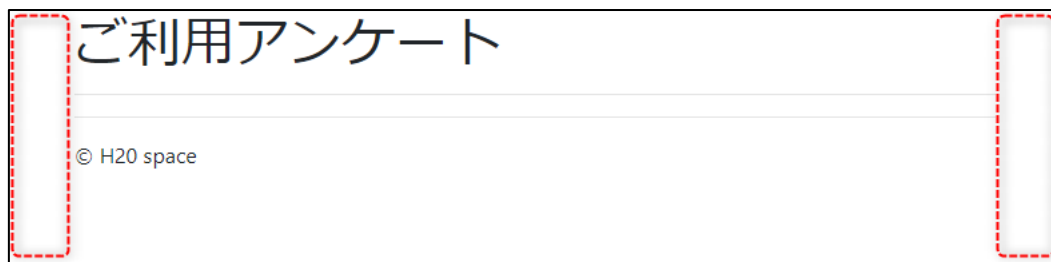
    <hr>

    <hr>

    <footer>
      <p>&copy; H20 space</p>
    </footer>
  </div> <!-- container-->
</body>

```

これで、画面の左右に余白が入りました（見出しやフッターは左寄せの状態です）。



Containers は、幅を狭めて中央に寄せるという効果があります。どのようなクラス名が使える、どのような機能があるかは Bootstrap のドキュメントを確認すると分かります。

- Containers <https://getbootstrap.com/docs/4.4/layout/overview/#containers>  
<https://getbootstrap.jp/docs/4.2/layout/overview/#containers> （日本語）

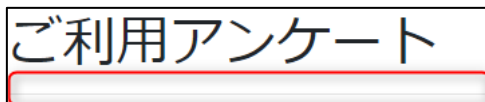
## 5. 見た目を調整する

Bootstrap は最低限のスタイル調整しか行いません。そのため、見出しなどは装飾されているというよりも、「ノーマライズされている」程度に留まっています。装飾などは自分で CSS を使って調整していきましょう。

まずは、見出しの見た目を整えていきます。

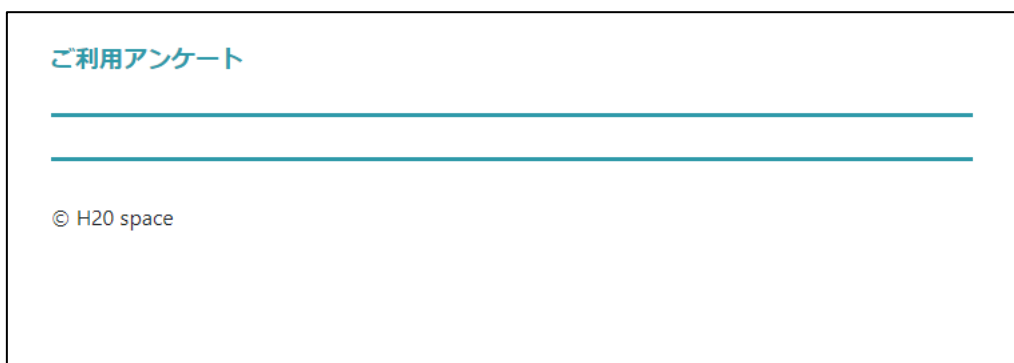


枠線は、<hr>タグを利用しました。<hr>は区切りを示す空要素で、横罫線などを引くのによく使われます。標準では、次のような水平線が引かれます。



「style.css」に次のように書き加えます。

```
header {  
  margin-top: 30px;  
  color: #2E99A9;  
}  
  
hr {  
  border-width: 3px;  
  border-color: #2E99A9;  
  margin: 30px 0;  
}  
  
h1 {  
  font-size: 18px;  
  font-weight: bold;  
  margin: 0;  
}
```



これで、ヘッダ部分が整いました。このように、文字の大きさや色などは、オリジナルの CSS で作り上げることができます。

ただし、ここでは水平線が画面いっぱいまで引かれず、途中で途切れてしまっています。

これは、親要素の「container」に幅が設定されてしまっているため。そこで、<hr>要素だけを「container」から外に出す必要があります。次のように HTML を変更しましょう。

```
<body>
  <div class="container">
    <header>
      <h1>ご利用アンケート</h1>
    </header>
  </div>
  <!-- container1-->

  <hr />

  <hr />

  <div class="container">
    <footer>
      <p>&copy; H20 space</p>
    </footer>
  </div>
  <!-- container2-->
</body>
```

Bootstrap は便利な反面、このように思わぬ部分でつまづくこともあるので注意しましょう。

ご利用アンケート
© H20 space

## 6. グリッドシステムを利用する

完成版のように画面の右端に「フォームへ戻る」リンクがあります。HTML に次のように記述しましょう。

```

<header>
  <h1>ご利用アンケート</h1>
  <a href="/">HOMEへ戻る</a>
</header>

```

これを、サイト名の右側に回り込ませるためには、これまでの知識では「float」プロパティを使って実現することができそうです。

しかし、Bootstrap を利用している場合は、それよりも「グリッドシステム」を活用すると良いでしょう。HTML を次のように変更します。

```

<div class="container">
  <header>
    <div class="row">
      <div class="col-sm-6">
        <h1>ご利用アンケート</h1>
      </div>
      <div class="col-sm-6">
        <a href="/">HOMEへ戻る</a>
      </div>
    </div>
  </header>
</div>
<!-- container1-->

```

これで、「float」プロパティを使ったときのようにリンクが回り込むようになりました。グリッドシステムは、画面を12個の「カラム」に分割し、利用したい幅の割合に応じてカラムを分け合って利用します。ここでは、6つずつのカラムに分けて、左右均等にしたというわけです。



グリッドシステムを使うときの書式は次の通りです。

## Bootstrap のグリッドシステムの書式

```
<div class="row">
  <div class="col-XX-X">
    ...
  </div>
  <div class="col-XX-X">
    ...
  </div>
  ...
</div>
```

まず、「row」というクラスの要素で囲みます。この中に要素を追加していき、1行が合計 12 カラムになるように調整します。ここでは 6 ずつのカラムに分けるため、次のようにしました。

col-sm-6

真ん中の「sm」は、レイアウトが切り替わるポイント（ブレイクポイント）の指定です。あとは、リンクだけ画面の右端に寄せれば完成です。ここでは「align-right」というクラス名を付けて、汎用的に右寄せにできるようにしましょう。

※「align-right」というクラス名はあまり好ましい名前ではありません。なぜなら、もし今後レイアウトの変更などで右寄せでなくなった場合、名前とつじつまが合わなくなるからです。

index.html

```
<div class="row">
  <div class="col-sm-6">
    <h1>ご利用アンケート</h1>
  </div>
  <div class="col-sm-6 align-right">
    <a href="/">HOMEへ戻る</a>
  </div>
</div>
```

style.css

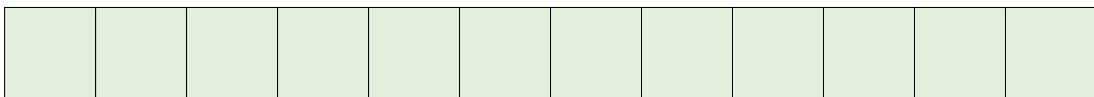
```
.align-right {
  text-align: right;
}
```

ご利用アンケート	HOMEへ戻る
© H20 space	

## 7. グリッドシステムとは

グリッドシステムは、現在の CSS フレームワークにはほとんど備わっている、画面レイアウト手法の 1 つです。

画面をいくつかの領域に分けて (Bootstrap の場合は、標準は 12 カラム)、何カラム文のスペースを利用するかを設定しながら、自由なレイアウトを行えるようになっています。リキッドレイアウトに対応していて、各カラムの幅が調整されます。



画面幅が狭くても自動的に調整される



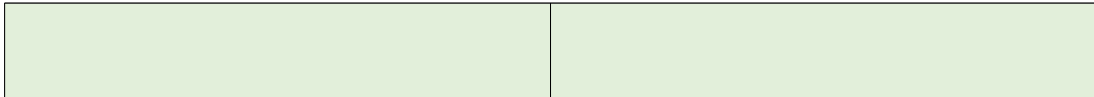
このカラムを、「使いたい幅」に合わせて使っていきます。たとえば本文のように、左右均等の 2 つのカラムにしたい場合は、次のように記述します。

```
<div class="row">
  <div class="col-md-6">
    ...
  </div>
```

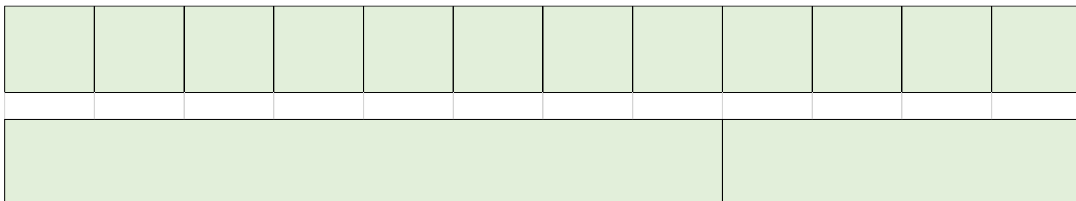
```

<div class="col-md-6">
  ...
</div>
</div> <!-- row -->

```



たとえば、サイドバーのように右側を少し狭いカラムにしたい場合は、8 と 4 などと調整します。



```

<div class="row">
  <div class="col-md-8">
    ...
  </div>
  <div class="col-md-4">
    ...
  </div>
</div> <!-- row -->

```

また、一時的に余白を作りたい場合は「オフセット」という仕組みを使うことができます。たとえば、左右に 2 カラム分ずつの余白を作る場合は、次のようにします。

```

<div class="row">
  <div class="col-md-8 offset-md-2">
    ...
  </div>
</div> <!-- row -->

```

ここでは、「offset-md-2」という記述によって、左側に2カラム分の余白ができました。そして、カラムの幅は8としているため、オフセット分を合わせても10となり、2カラム分余っています。このように余らせたカラムが右側の余白になります。


すこしややこしく感じますが、グリッドシステムを利用すると自由なレイアウトができる上、レスポンス Web デザインにも簡単に対応できます。使いこなしていきましょう。

## 8. グリッドシステムのレスポンス Web デザイン

グリッドシステムの便利な点がもう1つあります。Web ブラウザの幅を縮めて、768px 未満にしてみましょう。離れているので分かりにくいですが、回り込みが解除され、「ご利用アンケート」と「HOME へ戻る」が2行になって表示されます。先ほどの真ん中で指定した「sm」という記述がポイントです。

ご利用アンケート	HOMEへ戻る
© H20 space	

Bootstrap のグリッドシステムは、標準でレスポンス Web デザインに対応しています。そのブレイクポイント（RWD でレイアウトが変わるポイントとなる画面幅）を、「xs（768px 未満）」「sm（992px 未満）」「md（1200px 未満）」「lg（それ以上）」という4種類で定義しています。

xs	sm	md	lg
～767px	768～991px	992～1199px	1200px～

グリッドを使うときに、このうち、どのブレイクポイントに対応させるかを選ぶことができるのです。先ほど指定した、

```
<div class="col-sm-6">
```

という指定はつまり、画面幅が 768px までは 6 カラムにするという指定で、それ以下になった場合は標準に戻ります (12 カラムになります)。

次のように、複数のブレイクポイントに対応させることもできます。

```
<header>
  <div class="row">
    <div class="col-sm-6">
      <h1>ご利用アンケート</h1>
    </div>
    <div class="col-md-4 col-sm-8 col-xs-6 align-right">
      <a href="/">HOME へ戻る</a>
    </div>
  </div>
</header>
```

この場合、md までは 8 : 4、sm までは 4 : 8、そして xs までは 6 : 6 という画面幅によってレイアウトがころころ変わるといようなレイアウトを作ることができるのです。とはいえ、あまりやりすぎると利用者が混乱してしまうので気を付けましょう。

## 9. フォームを作成する <form>

それでは、フォームを作成していきます。説明を後にしていましたが、フォームの内容を送信するために、親要素として<form>要素が必要です。index.html に次のように追加します。



```
<div class="container">
  <form action="http://localhost/" method="post">

  </form>
</div>
```

<form>要素は、次のような書式で指定します。

```
<form action="送信先" method="get または post">
```

action 属性には、入力したフォームの送信先 (URL) を指定します。フォームの送信先には、送られた内容を処理するプログラムが必要となります。ここでは、送信先との連動部分は割愛します。

## 1. テキストフィールドを配置する

次に、フォームのパーツの1つであるテキストフィールドを配置します。

```
<form action="http://localhost/" method="post" class="row">
  <div class="col-sm-8 col-sm-offset-2">
    <p>この度は、ご利用いただきありがとうございます。恐れ入りますが、
    以下のフォームにご記入頂き、送信して頂けると幸いです。</p>
    <label for="name">お名前</label>
    <input type="text" id="name" name="name">
  </div> <!-- col-sm-8 -->
</form>
```

ここもグリッドシステムを使って幅を調整しました<form>要素に「row」という class 属性が付けられていることに注意しましょう。そして、<p>要素でフォームの説明を表示した後、フォーム本体を表示しています。

フォーム本体は、幅を少し狭くするために「col-sm-offset-2」を指定して、さらに幅自体も8としたため、両端に2カラムずつの余白を作っています。これで画面を表示すると次のようになります。

ご利用アンケート

HOMEへ戻る

この度は、ご利用いただきありがとうございます。恐れ入りますが、以下のフォームにご記入頂き、送信して頂けると幸いです。

お名前

© H2O space

<input>要素は次のような書式で記述します。

<input type="入力タイプ" name="フォーム名">

なお、「id」属性はグローバル属性です。type 属性に指定する「入力タイプ」には「text」のほか、入力する内容に従って、次のどれかから選びます。

- ・ text           一般的なテキスト
- ・ password     パスワード（入力した内容が隠れる）
- ・ email         メールアドレス
- ・ url            URL
- ・ file           ファイルパス（ユーザーの環境上のファイルを指定できる）
- ・ tel            電話番号
- ・ number        数値
- ・ date           日付（その他、datetime、datetime-local、month、time、week がある）
- ・ search        検索キーワード

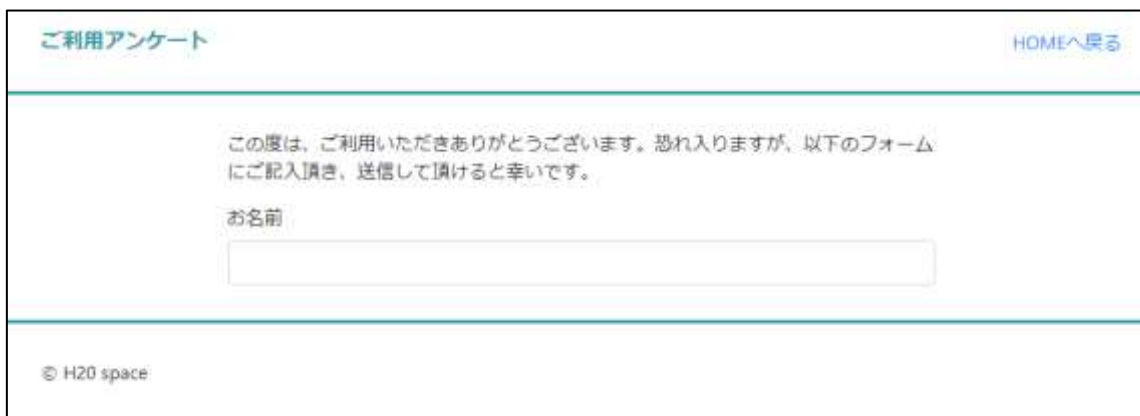
それぞれ、利用する Web ブラウザによって挙動は異なりますが、基本的には指定したものが入力しやすいように動作します。たとえば、スマートフォンで type 属性が「number」のテキストフィールドを入力すると、数字入力のキーボード画面が表示されます。なお、type 属性による挙動に対応していないブラウザでは、「text」を指定したのと同じ状態になります。

「name」属性は、フォームを送信したときにサーバサイドプログラムに渡される値の「名前」を指定します。他のパーツと名前が重複しないようにつける必要があります。通常は、サーバー側のプログラムによって指定などがあります。

## 2. Bootstrap を使ってスタイルを調整する

Bootstrap には、フォーム用のスタイルも準備されています。先ほど追加したテキストフィールドに「form-control」という class 属性を割り当て、<label>要素と<input>要素を「form-group」という class 属性の要素で囲みます。

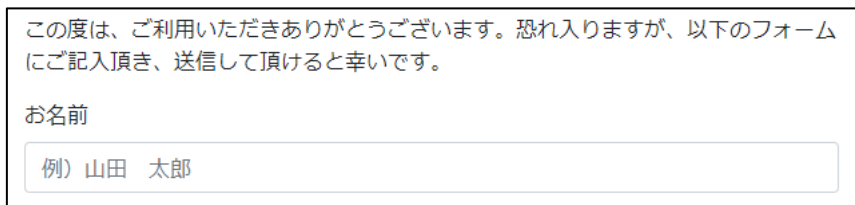
```
<div class="col-sm-8 offset-md-2">
  <p>この度は、ご利用いただきありがとうございます。恐れ入りますが、以下の
  <div class="form-group">
    <label for="name">お名前</label>
    <input type="text" id="name" name="name" class="form-control">
  </div>
</div> <!-- col-sm-8-->
```



## 3. 入力例を示す placeholder

<input>要素には、この他にもさまざまな属性を指定することができます。ここでは、入力例などを示す「placeholder」属性を使ってみましょう。

```
<input type="text" id="name" name="name" class="form-control" placeholder="例) 山田 太郎">
```



#### 4. ドロップダウンリストを設置する <select>と<option>

フォームには、テキストフィールド以外にもさまざまな入力欄があります。まずは、「ドロップダウンリスト」を利用しましょう。

```
<input type="text" id="name" name="name" class="form-control" />
</div>
<div class="form-group">
  <label for="job">ご職業</label>
  <select id="job" name="job" class="form-control">
    <option value="">選択してください</option>
    <option value="会社員">会社員</option>
    <option value="学生">学生</option>
    <option value="その他">その他</option>
  </select>
</div>
</div> <!-- col-sm-8-->
```

ドロップダウンリストの書式

```
<select name="パーツ名">
  <option value="値">表示ラベル</option>
  ...
</select>
```

必ず、<select>要素と<option>要素が親子で使われます。<option>要素は複数追加することができます。value 属性の値は、表示ラベルの内容とは異なる値を指定することもできます。

```
<option value=1>会社員</option>
```

```
<option value=1>学生</option>
```

こうすると、ユーザーが選択するときは「会社員」や「学生」と表示されますが、実際にフォームが送信されるときに、「1」や「2」といった下数字になって送信されます。サーバサイドプログラム側の仕様によって設定します。

## 5. 複数の選択ができるチェックボックスを設置する

続いて、チェックボックスを設置しましょう。1つだけを利用して「YES/NO」の選択肢を示すものや、複数を並べて選択できるものもあります。

規約に同意しますか？

☐ 同意します

「YES/NO」の選択肢を示す例

```
</select>
</div>
<div class="form-group">
  <label>知りたい内容（複数回答可）</label>
  <div class="checkbox">
    <label>
      <input type="checkbox" name="q1" value="html">
      HTML
    </label>
  </div>
  <div class="checkbox">
    <label>
      <input type="checkbox" name="q1" value="css">
      CSS
    </label>
  </div>
  <div class="checkbox">
    <label>
      <input type="checkbox" name="q1" value="javascript">
      JavaScript
    </label>
  </div>
</div>
</div> <!-- col-sm-8-->
```

チェックボックスの書式は次の通りです。

```
<input type="checkbox" name="パーツ名" value="送信する値">
```

チェックボックスは、タグを単体で利用すると四角いボックスしか表示されないため、その前後に必ず「ラベル」を付与します。特に Bootstrap では、ラベルを含めて<label>要素で囲むことになっています。さらにその上から、[checkbox]という class 属性のついた要素で囲みます。

チェックボックスを複数並べた場合、「name」属性がポイントとなります。name 属性は重複しないようにつけるのがルールでしたが、チェックボックスの場合、同じ設問項目の選択肢の場合には name 属性を同じ値にします。ここではいずれも「q1」としています。

## 6. 非常に重要な label の役割

<label>要素は、これまでも入力ボックスの説明（ラベル）としてマークアップしてきましたが、実はこの要素には重要な役割があります。

たとえば、「お名前」の部分をつまみかきまたはクリックしてみましょう。すると、テキストフィールドにテキストカーソルが表示されるのが分かります。（これを「フォーカスがあたり」といいます）。同じく、「ご職業」もつまみかき、クリックするとフォーカスがあたります。

## 7. 単一項目を選択するラジオボタンを設置する

チェックボックスと同じような入力ボックスで、1つしか選択できないのが「ラジオボタン」です。

```

        JavaScript
    </label>
</div>
</div>

<div class="form-group">
    <label>理解度はいかがですか？</label>
    <div>
        <label class="radio-inline">
            <input type="radio" name="q2" id="q2_1" value="1"> 理解できなかった
        </label>
        <label class="radio-inline">
            <input type="radio" name="q2" id="q2_2" value="2"> だいたい理解できた
        </label>
        <label class="radio-inline">
            <input type="radio" name="q2" id="q2_3" value="3"> 理解できた
        </label>
        <label class="radio-inline">
            <input type="radio" name="q2" id="q2_4" value="" checked> 回答しない
        </label>
    </div>
</div>
</div> <!-- col-sm-8-->

```

type 属性が「radio」になっている程度で、チェックボックスとほとんど同じです。

また Bootstrap では<label>要素に「radio-inline」クラスを付与すると、横に並べたときのスタイルが整います。

チェックボックスの場合でも同じように「checkbox-inline」クラスを付けることで、横に並べたときのスタイルが整います。

理解度はいかがですか？

☐ 理解できなかった
 ☐ だいたい理解できた
 ☐ 理解できた
 ☒ 回答しない

## 8. ラジオボタンを RWD 対応にする

横並びにしたラジオボタンは、そのままでは横幅の狭い画面で表示したとき、意図しないところで折り返されてしまいます。

知りたい内容（複数回答可）

☐ HTML

☐ CSS

☐ JavaScript

理解度はいかがですか？

☐ 理解できなかった ☐ だいたい理解できた ☐ 理解できた

☒ 回答しない

そこで、メディアクエリーを使って画面幅が狭くなったら縦並びになるようにしましょう。

```
@media only screen and (max-width: 992px) {
  .radio-inline {
    display: block;      /* ブロック要素にして改行されるように */
    margin: 0 !important; /* 余白を0に（強制的に） */
  }
}
```

画面幅が 992px 未満になったら、「radio-inline」クラスで指定されているスタイルを変更します。現座は「display」プロパティが「inline-block」になっているため、これを「block」に戻します。余白を付けるための「margin」プロパティがかかっているので、これも解除するために「0」を指定します。

しかし、この「margin」プロパティは後述する「優先順位の変更」によってそのままでは無効になってしまいます。そこで、「!important」をつけて強制的に適用させています。



知りたい内容（複数回答可）

☐ HTML
 ☐ CSS
 ☐ JavaScript

理解度はいかがですか？

☐ 理解できなかった
 ☐ だいたい理解できた
 ☐ 理解できた
 ☒ 回答しない

#### !important による優先順位の変更

CSS には優先順位があるため、新しくスタイルを記述してもうまく適用されないことがあります。そのような場合、確実に適用させたいプロパティに「!important」というオプションをつけると、優先順位が最高位になります。

多用すると、スタイルを整えにくくなりますので、注意しながら利用しましょう。

CSS の優先順位は以下のようになります。（下にいくほど優先順位が高くなります）

#### 9. 複数行の入力が可能なテキストエリア <textarea>

ご意見などを自由に記入できるエリアを追加します。<textarea>要素を使用します。この要素は、他のフォームパーツと異なり、空要素ではなくて閉じタグがあります。

```

<div class="form-group">
  <label for="message">ご意見</label>
  <textarea name="message" id="message" rows="10" class="form-control">
  </textarea>
</div>
</div> <!-- col-sm-8-->

```

理解度はいかがですか？

☐ 理解できなかった
 ☐ だいたい理解できた
 ☐ 理解できた
 ☒ 回答しない

ご意見

テキストエリアは次のような書式で記述します。

```
<textarea name="パーツ名" cols="値" rows="行数">初期値</textarea>
```

「cols」属性は、テキストエリアの「幅」を決める設定ですが、Bootstrap の場合はもともと「画面幅いっぱい」という設定がされているため、この属性指定は省略します。もし、Bootstrap を利用しない場合でも、レスポンス Web デザインを採用する場合は、「col」属性を指定することはできません。

なぜなら、テキストエリアの幅が固定されてしまい、スマートフォンの画面幅からはみ出してしまうためです。CSS の「width」属性で幅を設定するようにしましょう。

```
textarea {
  width: 100%;
}
```

## 10. 送信ボタンを設置する submit

最後に、送信ボタンは、<button>要素を使って設置します。

```
<button type="submit" class="btn btn-primary">送信する</button>
</div> <!-- col-sm-8-->
```

「btn btn-primary」クラスを付与すると Bootstrap のカスタムボタンを利用することがで

きます。( <https://getbootstrap.jp/docs/4.2/components/buttons/> )

#### リセットボタンの必要性

フォームには、送信ボタンの他に、リセットボタンを設置することができます。

```
<button type="reset" value="リセット">
```

このボタンをクリックすると、フォームの入力ボックスがすべて初期化されます。しかし、近年このボタンは設置しないのが一般的になってきています。それはたとえば、入会フォームや購入フォームなどで、「ここまで入力した内容をすべてなかったことにしたい」というケースがあまり想定できないうえに、送信ボタンと誤って押してしまった場合に、それまでの入力内容がすべて消えてしまい、リスクやデメリットの方が大きいからです。

#### 11. 必須項目を作る required

入力フォームには、必ず入力してほしい「必須項目」があります。この場合「required」属性を付けておけば、未入力の場合に警告を表示することができます。ここでは、「お名前」という入力項目を必須項目にしてみましょう。

```
<input type="text" id="name" name="name" class="form-control"
placeholder="例) 山田 太郎" required>
```

The screenshot shows a web form with three input fields. The first field is labeled 'お名前' (Name) and contains the placeholder text '例) 山田 太郎'. The second field is labeled 'ご職業' (Occupation) and has a dropdown menu with the text '選択してください'. The third field is labeled '知りたい内容 (複数回答可)' (Content you want to know (multiple answers possible)). A red-bordered box highlights the 'required' attribute in the code above. A warning message box with an exclamation mark icon and the text 'このフィールドを入力してください。' (Please enter this field.) is displayed over the 'ご職業' field.

しかし、属性を付加しただけでは、ユーザーにはどの項目が必須項目か分からないため、画面上にも明示するようにしましょう。<label>要素にも次のように追加します。

( <https://getbootstrap.jp/docs/4.2/components/badge/> )

```
<label for="name">お名前<span class="badge badge-danger">必須</span></label>
```