

### 3.3.3 画像の位置合わせ

画像を位置合わせするには、**Float ユーティリティ**(P.338 参照) または **Text ユーティリティ**(P.347 参照) を使用します。ブロックレベルの画像には、Spacing ユーティリティ (P.318 参照) の **mx-auto クラス**を使用して中央揃えにすることもできます。

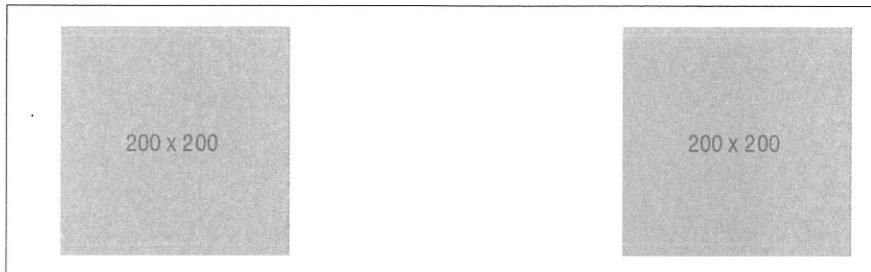
#### ■ Float ユーティリティによる位置合わせ

次の例では、画像の位置合わせに Float ユーティリティを使用しています (リスト 3-51、図 3-29)。最初の img 要素に **float-left クラス**を追加して画像を左に配置、次の img 要素には **float-right クラス**を追加して画像を右に配置しています。

▼リスト 3-51 Float ユーティリティを使用した配置 (img-float.html)

```
<div class="clearfix">
  
  
</div>
```

▼図 3-29 Float ユーティリティクラスを使用した配置例



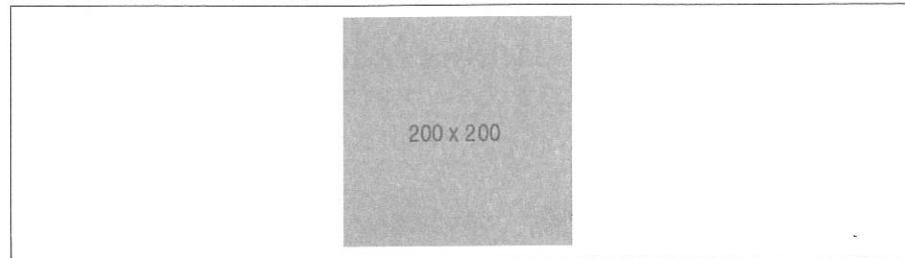
#### ■ Text ユーティリティによる位置合わせ

次の例では、画像の位置合わせに Text ユーティリティを使用し、img 要素に **text-center クラス**を追加して画像を中央揃えに配置しています (リスト 3-52、図 3-30)。

▼リスト 3-52 text-center クラスで中央揃えに配置 (img-text.html)

```
<div class="text-center">
  
</div>
```

▼図 3-30 text-center クラスで中央揃えに配置



## ■ Spacing ユーティリティによる位置合わせ

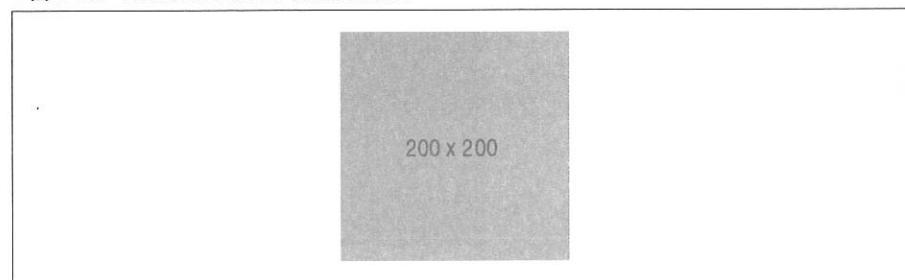
次の例では、Display ユーティリティ（P.310 参照）の **d-block クラス**でブロックレベル化された img 要素に、Spacing ユーティリティ（P.318 参照）の **mx-auto クラス**を追加して、画像を中央揃えに配置しています（リスト 3-53、図 3-31）。

▼リスト 3-53 mx-auto クラスを使用した配置（img-spacing.html）

```

```

▼図 3-31 mx-auto クラスで中央揃えに配置



NOTE

### picture 要素

picture 要素は HTML5.1 以降で新しく加わった、レスポンシブ画像を扱うための要素です。img 要素と source 要素を組み合わせて、ビューポートや画面幅などに応じて複数の画像を出し分けることができます。Bootstrap では、picture 要素に直接 **img-fluid クラス**や **img-thumbnail クラス**を追加することができません。これらのクラスの追加が必要な場合は、子要素である img 要素に追加するようにしましょう（リスト 3-54）。

▼リスト 3-54 picture 要素に直接 img-\* クラスを追加しない

```
<picture>
  <source srcset="..." type="image/svg+xml">
  
</picture>
```

3

SECTION

## 4

## テーブル

1  
2  
3  
4  
5  
6  
7  
8  
9  
10

**table** 要素は、カレンダーや日付選択ツールなどのサードパーティのウィジェットでも広く使用されています。そのため、Bootstrap でのテーブルスタイルはオプトイン（明示的に許諾する方法）で使えるように設計されています。任意の **table** 要素に対してオプトインで **table クラス** を追加してから、さまざまな定義済みクラスを加えてバリエーションを拡張していきます。

### 3.4.1 テーブルの基本スタイリング

**table** 要素に **table クラス** を追加して、Bootstrap での基本的なテーブルをスタイリングします。ネストされたテーブルには、親テーブルと同じスタイルが継承されます（リスト 3-55、図 3-32）。

▼リスト 3-55 テーブルの基本スタイル（table-basic.html）

```
<table class="table">
  <thead>
    <tr>
      <th scope="col">見出しへル</th>
      <th scope="col">見出しへル</th>
      <th scope="col">見出しへル</th>
      <th scope="col">見出しへル</th>
    </tr>
  </thead>
  <tbody>
    <tr>
      <th scope="row">見出しへル</th>
      <td>データセル</td>
      <td>データセル</td>
      <td>データセル</td>
    </tr>
    <tr>
      <th scope="row">見出しへル</th>
      <td>データセル</td>
      <td>データセル</td>
      <td>データセル</td>
    </tr>
    <tr>
      <th scope="row">見出しへル</th>
      <td>データセル</td>
      <td>データセル</td>
      <td>データセル</td>
    </tr>
  </tbody>

```

```
<td>データセル</td>
</tr>
</tbody>
</table>
```

▼図 3-32 テーブルの基本スタイル

|       |       |       |       |
|-------|-------|-------|-------|
| 見出しセル | 見出しセル | 見出しセル | 見出しセル |
| 見出しセル | データセル | データセル | データセル |
| 見出しセル | データセル | データセル | データセル |
| 見出しセル | データセル | データセル | データセル |

なお **table クラス**には、リスト 3-56 のようなスタイルで幅やマージンサイズ、背景色（透明）が定義されています。コンテンツエリアに対して、常に 100% の幅になるよう、レスポンシブなスタイルになっています。

▼リスト 3-56 table クラスに定義されているスタイル

```
.table {
  width: 100%;
  max-width: 100%;
  margin-bottom: 1rem;
  background-color: transparent;
}
```

さらに、table クラスの子孫要素の th 要素や td 要素、tbody 要素には、リスト 3-57 のようなスタイルでパディングやボーダーなどが設定されており、テーブルの内容が見やすくなるよう調整されています。

▼リスト 3-57 table クラスの子孫要素に定義されているスタイル

```
.table th,
.table td {
  padding: 0.75rem;
  vertical-align: top;
  border-top: 1px solid #dee2e6;
}

.table thead th {
  vertical-align: bottom;
  border-bottom: 2px solid #dee2e6;
}

.table tbody + tbody {
  border-top: 2px solid #dee2e6;
```

```

}
.table .table {
  background-color: #fff;
}

```

### 3.4.2 暗色テーブル

table クラスが設定された table 要素に **table-dark** クラスを追加することで、表示色の明暗を反転させることができます（リスト 3-58、図 3-33）。

▼リスト 3-58 暗色テーブル（table-dark.html）

```

<table class="table table-dark">
  ...
</table>

```

▼図 3-33 暗色テーブル

| 見出しセル | 見出しセル | 見出しセル | 見出しセル |
|-------|-------|-------|-------|
| 見出しセル | データセル | データセル | データセル |
| 見出しセル | データセル | データセル | データセル |
| 見出しセル | データセル | データセル | データセル |

なお **table-dark** クラスには、暗めの背景色と明るめの文字色が定義されています（リスト 3-59）。

▼リスト 3-59 table-dark クラスに定義されているスタイル

```

.table-dark,
.table-dark > th,
.table-dark > td {
  background-color: #c6c8ca;
}
...
.table-dark {
  color: #fff;
  background-color: #212529;
}
.table-dark th,
.table-dark td,
.table-dark thead th {
  border-color: #32383e;
}

```

### 3.4.3 テーブルヘッドのオプション

thead要素に**thead-dark**クラス、**thead-light**クラスを追加して、thead要素の表示色の明暗を変更することができます（リスト3-60、図3-34）。

▼リスト3-60 thead-darkクラスとthead-lightクラス（table-head-option.html）

```
<!-- [thead-dark] -->
<table class="table">
  <thead class="thead-dark">
    ...中略...
  </thead>
  <tbody>
    ...中略...
  </tbody>
</table>

<!-- [thead-light] -->
<table class="table">
  <thead class="thead-light">
    ...中略...
  </thead>
  <tbody>
    ...中略...
  </tbody>
</table>
```

▼図3-34 thead-darkクラスとthead-lightクラス

| 見出しセル | 見出しセル | 見出しセル | 見出しセル |
|-------|-------|-------|-------|
| 見出しセル | データセル | データセル | データセル |
| 見出しセル | データセル | データセル | データセル |
| 見出しセル | データセル | データセル | データセル |
| 見出しセル | 見出しセル | 見出しセル | 見出しセル |
| 見出しセル | データセル | データセル | データセル |
| 見出しセル | データセル | データセル | データセル |
| 見出しセル | データセル | データセル | データセル |

なお**thead-dark**クラスには暗めの背景色と明るめの文字色、**thead-light**クラスには明るめの背景色と暗めの文字色が定義されています（リスト3-61）。

▼リスト 3-61 thead-dark クラスと thead-light クラスに定義されているスタイル

```
.table .thead-dark th {
  color: #fff;
  background-color: #212529;
  border-color: #32383e;
}

.table .thead-light th {
  color: #495057;
  background-color: #e9ecef;
  border-color: #e9ecef;
}
```

### 3.4.4 縞模様のテーブル

table クラスが設定された table 要素に **table-striped クラス**を追加して、テーブルを縞模様にすることができます（リスト 3-62、図 3-35）。

▼リスト 3-62 table-striped クラスを使用した縞模様のテーブル（table-striped.html）

```
<table class="table table-striped">
…中略…
</table>
```

▼図 3-35 table-striped クラスを使用した縞模様のテーブル

|       |       |       |       |
|-------|-------|-------|-------|
| 見出しセル | 見出しセル | 見出しセル | 見出しセル |
| 見出しセル | データセル | データセル | データセル |
| 見出しセル | データセル | データセル | データセル |
| 見出しセル | データセル | データセル | データセル |
| 見出しセル | 見出しセル | 見出しセル | 見出しセル |
| 見出しセル | データセル | データセル | データセル |
| 見出しセル | データセル | データセル | データセル |
| 見出しセル | データセル | データセル | データセル |

なお **table-striped クラス**には、リスト 3-63 のようなスタイルで、奇数行の背景色が不透明度 5% の黒になるような設定が定義されています（①）。また暗色テーブルでの明暗の反転を表現するために、奇数行の背景が不透明度 5% の白になるような設定が定義されています（②）。

## ▼リスト 3-63 table-striped クラスに定義されているスタイル

```
.table-striped tbody tr:nth-of-type(odd) {
  background-color: rgba(0, 0, 0, 0.05); —————①
}
…中略…
.table-dark.table-striped tbody tr:nth-of-type(odd) {
  background-color: rgba(255, 255, 255, 0.05); —————②
}
```

**3.4.5 畳線付きのテーブル**

table クラスが設定された table 要素に **table-bordered クラス**を追加して、テーブルに畳線を表示することができます（リスト 3-64、図 3-36）。

## ▼リスト 3-64 table-bordered クラスを使用した畳線付きテーブル（table-bordered.html）

```
<table class="table table-bordered">
…中略…
</table>
```

## ▼図 3-36 table-bordered クラスを使用した畳線付きテーブル

|       |       |       |       |
|-------|-------|-------|-------|
| 見出しセル | 見出しセル | 見出しセル | 見出しセル |
| 見出しセル | データセル | データセル | データセル |
| 見出しセル | データセル | データセル | データセル |
| 見出しセル | データセル | データセル | データセル |
| 見出しセル | 見出しセル | 見出しセル | 見出しセル |
| 見出しセル | データセル | データセル | データセル |
| 見出しセル | データセル | データセル | データセル |
| 見出しセル | データセル | データセル | データセル |

**table-bordered クラス**には、リスト 3-65 のようなスタイルが定義されており、テーブルには 1px 幅のボーダー、見出しセルの下辺にのみ 2px 幅のボーダーがスタイルリングされ、視覚的に見出しセルがわかりやすく表現されています。

## ▼リスト 3-65 table-bordered クラスに定義されているスタイル

```
.table-bordered {
  border: 1px solid #dee2e6;
}
.table-bordered th,
.table-bordered td {
  border: 1px solid #dee2e6;
}
.table-bordered thead th,
.table-bordered thead td {
  border-bottom-width: 2px;
}
…中略…
.table-dark.table-bordered {
  border: 0;
}
```

**3.4.6 罫線なしのテーブル 4.1**

table クラスが設定された table 要素に **table-borderless クラス**を追加すると、罫線なしのテーブルになります（リスト 3-66、図 3-37）。

## ▼リスト 3-66 table-borderless クラスを使用した罫線なしテーブル（table-borderless.html）

```
<table class="table table-borderless">
…中略…
</table>
```

▼図 3-37 table-borderless クラスを使用した罫線なしテーブル

|       |       |       |       |
|-------|-------|-------|-------|
| 見出しセル | 見出しセル | 見出しセル | 見出しセル |
| 見出しセル | データセル | データセル | データセル |
| 見出しセル | データセル | データセル | データセル |
| 見出しセル | データセル | データセル | データセル |
| 見出しセル | 見出しセル | 見出しセル | 見出しセル |
| 見出しセル | データセル | データセル | データセル |
| 見出しセル | データセル | データセル | データセル |
| 見出しセル | データセル | データセル | データセル |

**table-borderless クラス**には、リスト 3-67 のようなスタイルが定義されており、th や td、tbody 要素のボーダーが 0 に設定されています。

▼リスト 3-67 table-borderless クラスに定義されているスタイル

```
.table-borderless th,  
.table-borderless td,  
.table-borderless thead th,  
.table-borderless tbody + tbody {  
    border: 0;  
}
```

### 3.4.7 テーブル行のマウスオーバー表示

table クラスが設定された table 要素に **table-hover クラス**を追加して、tbody 要素内のテーブル行にマウスオーバー表示を設定することができます（リスト 3-68、図 3-38）。

▼リスト 3-68 table-hover クラスを使用したマウスオーバー表示（table-hover.html）

```
<table class="table table-hover">  
…中略…  
</table>
```

▼図 3-38 table-hover クラスを使用したマウスオーバー表示

|       |       |       |       |
|-------|-------|-------|-------|
| 見出しセル | 見出しセル | 見出しセル | 見出しセル |
| 見出しセル | データセル | データセル | データセル |
| 見出しセル | データセル | データセル | データセル |
| 見出しセル | データセル | データセル | データセル |
| 見出しセル | データセル | 見出しセル | 見出しセル |
| 見出しセル | データセル | データセル | データセル |
| 見出しセル | データセル | データセル | データセル |
| 見出しセル | データセル | データセル | データセル |

なお **table-hover クラス**には、リスト 3-69 のようなスタイルで、マウスオーバー表示の背景色が不透明度 7.5 %の黒に設定されるように定義されています。

▼リスト 3-69 **table-hover** クラスに定義されているスタイル

```
.table-hover tbody tr:hover {
  background-color: rgba(0, 0, 0, 0.075);
}
```

**3.4.8 テーブルのコンパクト化**

table クラスが設定された table 要素に **table-sm** クラスを追加して、セルのパディングサイズを小さくし、テーブルをコンパクト化することができます（リスト 3-70、図 3-39）。

▼リスト 3-70 **table-sm** クラスを使用したコンパクトなテーブル（table-sm.html）

```
<table class="table table-sm">
  ...
</table>
```

▼図 3-39 **table-sm** クラスを使用したコンパクトなテーブル

|       |       |       |       |
|-------|-------|-------|-------|
| 見出しセル | 見出しセル | 見出しセル | 見出しセル |
| 見出しセル | データセル | データセル | データセル |
| 見出しセル | データセル | データセル | データセル |
| 見出しセル | データセル | データセル | データセル |
| 見出しセル | 見出しセル | 見出しセル | 見出しセル |
| 見出しセル | データセル | データセル | データセル |
| 見出しセル | データセル | データセル | データセル |
| 見出しセル | データセル | データセル | データセル |

なお **table-sm** クラスには、リスト 3-71 のようなスタイルでパディングサイズが定義されています。

▼リスト 3-71 **table-sm** クラスに定義されているスタイル

```
.table-sm th,
.table-sm td {
  padding: 0.3rem;
}
```

### 3.4.9 テーブル行・セルの色付け

テーブルの行やセルの要素に背景色を指定するクラスを追加して、色付けを行うことができます。背景色用クラスの詳細は、表3-3を参照してください。

▼表3-3 表 テーブルの背景用クラス

| 背景色       | クラス             | 色                    | 色（マウスオーバー行）          |
|-----------|-----------------|----------------------|----------------------|
| Active    | table-active    | rgba(0, 0, 0, 0.075) | rgba(0, 0, 0, 0.075) |
| Primary   | table-primary   | #b8daff              | #9fcdf               |
| Secondary | table-secondary | #d6d8db              | #c8cbcf              |
| Success   | table-success   | #c3e6cb              | #b1dfbb              |
| Danger    | table-danger    | #f5c6cb              | #f1b0b7              |
| Warning   | table-warning   | #ffeeba              | #ffe8a1              |
| Info      | table-info      | #bee5eb              | #abddde5             |
| Light     | table-light     | #fdfdfe              | #ecefef6             |
| Dark      | table-dark      | #32383e              | #b9bbbe              |

#### ■ テーブル行に色を付ける

テーブル行に色付けを行う場合は、tr要素に背景色用クラスを追加します（リスト3-72、図3-40）。

▼リスト3-72 テーブル行に色を付ける（table-bg-tr.html）

```
<table class="table">
  <thead>
    <tr>
      <th scope="col">背景色用のクラス</th>
      <th scope="col">見出しほれ</th>
      <th scope="col">見出しほれ</th>
      <th scope="col">見出しほれ</th>
    </tr>
  </thead>
  <tbody>
    <!-- 背景色：なし -->
    <tr>
      <th scope="row">なし（標準）</th>
      ...中略...
    </tr>
    <!-- 背景色：Active -->
    <tr class="table-active">
      <th scope="row">table-active</th>
      ...中略...
    </tr>
    <!-- 背景色：Primary -->
    <tr class="table-primary">
      <th scope="row">.table-primary</th>
    </tr>
  </tbody>
</table>
```

```
...中略...
</tr>
<!-- 背景色 : Secondary -->
<tr class="table-secondary">
    <th scope="row">table-secondary</th>
    ...中略...
</tr>
<!-- 背景色 : Success -->
<tr class="table-success">
    <th scope="row">table-success</th>
    ...中略...
</tr>
<!-- 背景色 : Danger -->
<tr class="table-danger">
    <th scope="row">table-danger</th>
    ...中略...
</tr>
<!-- 背景色 : Warning -->
<tr class="table-warning">
    <th scope="row">table-warning</th>
    ...中略...
</tr>
<!-- 背景色 : Info -->
<tr class="table-info">
    <th scope="row">table-info</th>
    ...中略...
</tr>
<!-- 背景色 : Light -->
<tr class="table-light">
    <th scope="row">table-light</th>
    ...中略...
</tr>
<!-- 背景色 : Dark -->
<tr class="table-dark">
    <th scope="row">table-dark</th>
    ...中略...
</tr>
</tbody>
</table>
```

1

2

3

4

5

6

7

8

9

10

▼図3-40 テーブル行に色を付ける

| 背景色クラス          | 見出しセル | 見出しセル | 見出しセル |
|-----------------|-------|-------|-------|
| なし(標準)          | データセル | データセル | データセル |
| table-active    | データセル | データセル | データセル |
| table-primary   | データセル | データセル | データセル |
| table-secondary | データセル | データセル | データセル |
| table-success   | データセル | データセル | データセル |
| table-danger    | データセル | データセル | データセル |
| table-warning   | データセル | データセル | データセル |
| table-info      | データセル | データセル | データセル |
| table-light     | データセル | データセル | データセル |
| table-dark      | データセル | データセル | データセル |

### 個々のセルに色を付ける

個々のセルに色付けを行う場合は、th要素またはtd要素に背景色用クラスを追加します(リスト3-73、図3-41)。

▼リスト3-73 個々のセルに色を付ける(table-bg-td.html)

```
<table class="table">
  <tr>
    <th>th(標準)</th>
    <th class="table-active">th.table-active</th>
  </tr>
  <tr>
    <!-- 背景色: Primary -->
    <td class="table-primary">td.table-primary</td>
    <!-- 背景色: Secondary -->
    <td class="table-secondary">td.table-secondary</td>
  </tr>
  <tr>
    <!-- 背景色: Success -->
    <td class="table-success">td.table-success</td>
    <!-- 背景色: Danger -->
    <td class="table-danger">td.table-danger</td>
  </tr>
  <tr>
    <!-- 背景色: Warning -->
  </tr>
```

```
<td class="table-warning">td.table-warning</td>
<!-- 背景色 : Info -->
<td class="table-info">td.table-info</td>
</tr>
<tr>
  <!-- 背景色 : Light -->
  <td class="table-light">td.table-light </td>
  <!-- 背景色 : Dark -->
  <td class="table-dark">td.table-dark</td>
</tr>
</table>
```

▼図 3-41 個々のセルに色を付ける

| th (標準)          | th.table-active    |
|------------------|--------------------|
| td.table-primary | td.table-secondary |
| td.table-success | td.table-danger    |
| td.table-warning | td.table-info      |
| td.table-light   | td.table-dark      |

1  
2  
3  
4  
5  
6  
7  
8  
9  
10

## Color ユーティリティで色を付ける

Color ユーティリティ (P.302 参照) の **bg-{ 色の種類 } クラス**を追加して、テーブル行または個々のセルに色付けを行うこともできます。標準のテーブル背景色を使用できない暗色テーブルにも色付けを行うことができます (リスト 3-74、図 3-42)。

▼リスト 3-74 暗色テーブルに Color ユーティリティで色を付ける (table-bg-dark.html)

```
<!-- Color ユーティリティでテーブル行に色を付ける -->
<table class="table table-dark">
  <thead>
    <tr>
      <th scope="col">Color ユーティリティ</th>
      ...中略...
    </tr>
  </thead>
  <tbody>
    <tr>
      <th scope="row">なし (標準)</th>
      ...中略...
    </tr>
    <!-- bg-primary -->
```

```
<tr class="bg-primary">
  <th scope="row">bg-primary </th>
  ...中略...
</tr>
<!-- bg-primary -->
<tr class="bg-primary">
  <th scope="row">bg-success </th>
  ...中略...
</tr>
<!-- bg-warning -->
<tr class="bg-warning">
  <th scope="row">bg-warning </th>
  ...中略...
</tr>
<!-- bg-danger -->
<tr class="bg-danger">
  <th scope="row">bg-danger </th>
  ...中略...
</tr>
<!-- bg-info -->
<tr class="bg-info">
  <th scope="row">bg-info </th>
  ...中略...
</tr>
</tbody>
</table>

<!-- Colorユーティリティで個々のセルに色を付ける -->
<table class="table table-dark">
  <tr>
    <th>th（標準） </th>
    <!-- bg-primary -->
    <th class="bg-primary">th.bg-primary</th>
  </tr>
  <tr>
    <!-- bg-success -->
    <td class="bg-success">td.bg-success</td>
    <!-- bg-warning -->
    <td class="bg-warning">td.bg-warning</td>
  </tr>
  <tr>
    <!-- bg-danger -->
    <td class="bg-danger">td.bg-danger </td>
    <!-- bg-info -->
    <td class="bg-info">td.bg-info</td>
  </tr>
</table>
```

▼図 3-42 Color ユーティリティで色を付ける

| Colorユーティリティ            | 見出しセル | 見出しセル | 見出しセル |
|-------------------------|-------|-------|-------|
| なし(標準)                  | データセル | データセル | データセル |
| <code>bg-primary</code> | データセル | データセル | データセル |
| <code>bg-success</code> | データセル | データセル | データセル |
| <code>bg-warning</code> | データセル | データセル | データセル |
| <code>bg-danger</code>  | データセル | データセル | データセル |
| <code>bg-info</code>    | データセル | データセル | データセル |

| th (標準)                    | th.bg-primary              |
|----------------------------|----------------------------|
| <code>td.bg-success</code> | <code>td.bg-warning</code> |
| <code>td.bg-danger</code>  | <code>td.bg-info</code>    |

### 3.4.10 キャプション

Bootsrap で定義済みのスタイルを使用して、**caption** 要素を表示します。caption 要素はテーブルのキャプションの役割を果たし、スクリーンリーダーのユーザーが、そのテーブルが何であるかを理解し、どう読むかを決定するのに役立ちます（リスト 3-75、図 3-43）。

▼リスト 3-75 caption 要素 (table-caption.html)

```
<table class="table">
  <caption>このテーブルのキャプション</caption>
  <thead>
    ...中略...
  </thead>
  <tbody>
    ...中略...
  </tbody>
</table>
```

▼図 3-43 caption 要素

|               |       |       |       |
|---------------|-------|-------|-------|
| 見出しセル         | 見出しセル | 見出しセル | 見出しセル |
| 見出しセル         | データセル | データセル | データセル |
| 見出しセル         | データセル | データセル | データセル |
| 見出しセル         | データセル | データセル | データセル |
| このテーブルのキャプション |       |       |       |

なお Bootstrap における caption 要素には、リスト 3-76 のようなスタイルでパディングサイズや文字設定が定義されています。また、キャプションの表示位置がテーブルの下に設定されるように定義されています（❶）。

▼リスト 3-76 caption 要素に定義されているスタイル

```
caption {
  padding-top: 0.75rem;
  padding-bottom: 0.75rem;
  color: #6c757d;
  text-align: left;
  caption-side: bottom; ──────────────────────────❶
}
```

### 3.4.11 レスポンシブ対応のテーブル

Bootstrap では、テーブルを水平方向にスクロール可能にすることでレスポンシブ対応させます。table 要素に **table クラス** および **table-responsive クラス** を追加して、すべてのビューポートに対応するテーブルを作成します（リスト 3-77）。

▼リスト 3-77 レスポンシブ対応テーブルと標準テーブルとの比較（table-responsive.html）

```
<!-- レスポンシブ対応テーブル -->
<div class="table-responsive">
  <table class="table">
    <caption>
      レスポンシブ対応テーブル (table-responsive)
    </caption>
    <thead>
      …中略…
    </thead>
    <tbody>
      …中略…
    </tbody>
  </table>
</div>
```

```

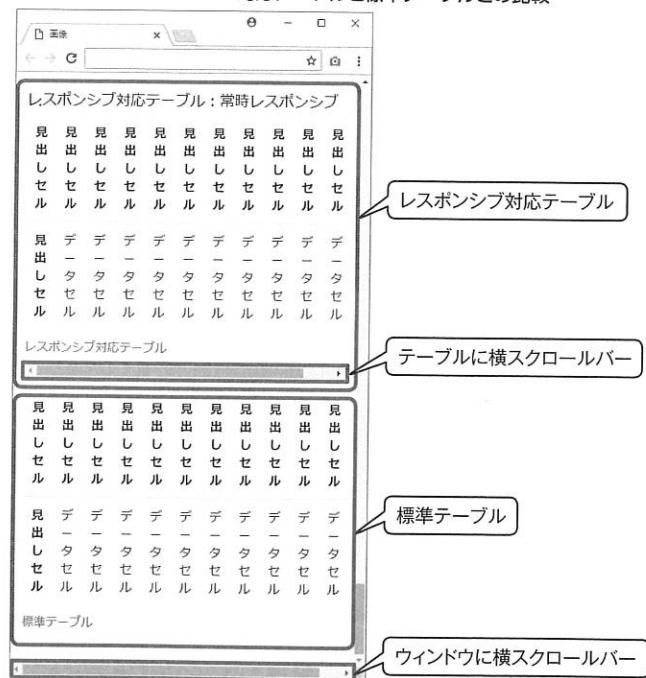
</table>
</div>

<!-- 標準テーブル -->
<table class="table">
  <caption>
    標準テーブル (table)
  </caption>
  <thead>
    ...中略...
  </thead>
  <tbody>
    ...中略...
  </tbody>
</table>

```

親要素に **table-responsive** クラスを追加されたテーブルは、画面幅がテーブルより小さい場合に横スクロールバーが表示され、セルを閲覧することができます。一方、標準のテーブルは、テーブルの右端が表示されず、ウィンドウに横スクロールバーが現れます（図 3-44）。

▼図 3-44 レスポンシブ対応テーブルと標準テーブルとの比較



1  
2  
3  
4  
5  
6  
7  
8  
9  
10

なお **table-responsive クラス**には、リスト 3-78 のようなスタイルで表示形式や幅などが定義されています。また Webkit 系の端末（Android、iOS など）でスムーススクロールを設定し（①）、Internet Explorer でスクロールバーを自動的に隠し、スクロールバーがコンテンツに重なる状態を回避するような設定（②）が定義されています。

▼リスト 3-78 table-responsive クラスに定義されているスタイル

```
.table-responsive {
  display: block;
  width: 100%;
  overflow-x: auto;
  -webkit-overflow-scrolling: touch; ①
  -ms-overflow-style: -ms-autohiding-scrollbar; ②
}
```

ブレイクポイントに応じたレスポンシブ対応テーブルを作成する場合は、テーブルの親要素に **table-responsive-{ ブレイクポイント } クラス**を追加します。

リスト 3-79 のサンプルのように、親要素に **table-responsive-sm クラス**を持つテーブルであれば、画面幅が sm 未満でテーブルの幅がコンテナよりも広い場合に、横スクロールバーが現れます（図 3-45）。

▼リスト 3-79 ブレイクポイントに応じたレスポンシブ対応テーブル（table-responsive-breakpoint.html）

```
<!-- レスポンシブ対応テーブル：画面幅sm未満 -->
<div class="table-responsive-sm">
  <table class="table">
    ...中略...
  </table>
</div>

<!-- レスポンシブ対応テーブル：画面幅md未満 -->
<div class="table-responsive-md">
  <table class="table">
    ...中略...
  </table>
</div>

<!-- レスポンシブ対応テーブル：画面幅lg未満 -->
<div class="table-responsive-lg">
  <table class="table">
    ...中略...
  </table>
</div>

<!-- レスポンシブ対応テーブル：画面幅xl未満 -->
<div class="table-responsive-xl">
  <table class="table">
    ...中略...
  </table>
</div>
```

```
</table>
</div>
```

▼図 3-45 ブレイクポイントに応じたレスポンシブ対応テーブル（画面幅 sm）

レスポンシブ対応テーブル：画面幅sm以下

|   |   |   |   |   |   |   |   |   |   |   |   |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 見 | 見 | 見 | 見 | 見 | 見 | 見 | 見 | 見 | 見 | 見 | 見 |
| 出 | 出 | 出 | 出 | 出 | 出 | 出 | 出 | 出 | 出 | 出 | 出 |
| し | し | し | し | し | し | し | し | し | し | し | し |
| セ | セ | セ | セ | セ | セ | セ | セ | セ | セ | セ | セ |
| ル | ル | ル | ル | ル | ル | ル | ル | ル | ル | ル | ル |
| 見 | デ | デ | デ | デ | デ | デ | デ | デ | デ | デ | デ |
| 出 | ー | ー | ー | ー | ー | ー | ー | ー | ー | ー | ー |
| し | タ | タ | タ | タ | タ | タ | タ | タ | タ | タ | タ |
| セ | セ | セ | セ | セ | セ | セ | セ | セ | セ | セ | セ |
| ル | ル | ル | ル | ル | ル | ル | ル | ル | ル | ル | ル |

レスポンシブ対応テーブル  
(画面幅sm以下)

レスポンシブ対応テーブル  
(画面幅 sm 未満)  
テーブルに横スクロールバー

レスポンシブ対応テーブル：画面幅md以下

|   |   |   |   |   |   |   |   |   |   |   |   |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 見 | 見 | 見 | 見 | 見 | 見 | 見 | 見 | 見 | 見 | 見 | 見 |
| 出 | 出 | 出 | 出 | 出 | 出 | 出 | 出 | 出 | 出 | 出 | 出 |
| し | し | し | し | し | し | し | し | し | し | し | し |
| セ | セ | セ | セ | セ | セ | セ | セ | セ | セ | セ | セ |
| ル | ル | ル | ル | ル | ル | ル | ル | ル | ル | ル | ル |
| 見 | デ | デ | デ | デ | デ | デ | デ | デ | デ | デ | デ |
| 出 | ー | ー | ー | ー | ー | ー | ー | ー | ー | ー | ー |
| し | タ | タ | タ | タ | タ | タ | タ | タ | タ | タ | タ |
| セ | セ | セ | セ | セ | セ | セ | セ | セ | セ | セ | セ |
| ル | ル | ル | ル | ル | ル | ル | ル | ル | ル | ル | ル |

レスポンシブ対応テーブル  
(画面幅md以下)

レスポンシブ対応テーブル  
(画面幅 md 未満)

なお **table-responsive-{ ブレイクポイント } クラス**には、リスト 3-74 のようなスタイルで表示形式や幅、スクロール時の挙動などが定義されています。Bootstrap はモバイルファーストで設計されているため、基本的には @media (min-width: 576px) {...} というように指定の画面幅以上（576px 以上）の形式で、小さい画面幅を基準にして、そこから徐々に大きい画面幅のスタイルを上書きで指定しますが、ここでは @media (max-width: 565.98px) {...} というように、指定の画面幅以下（576px 未満）の形式で指定し、逆方向に進むメディアエクスリが使用されています（①）。また Webkit 系の端末（Android、iOS など）でスムーススクロールを設定し（②）、Internet Explorer でスクロールバーを自動的に隠し、スクロールバーがコンテンツに重なる状態を回避するような設定（③）が定義されています（リスト 3-80）。

▼リスト 3-80 table-responsive-{ ブレイクポイント } クラスに定義されているスタイル

```
@media (max-width: 575.98px) {
  .table-responsive-sm {
    display: block;
    width: 100%;
    overflow-x: auto;
    -webkit-overflow-scrolling: touch; ❶
    -ms-overflow-style: -ms-autohiding-scrollbar; ❷
  }
  .table-responsive-sm > .table-bordered {
    border: 0;
  }
}
```

```
        }
    }

@media (max-width: 767.98px) { ①
    .table-responsive-md {
        display: block;
        width: 100%;
        overflow-x: auto;
        -webkit-overflow-scrolling: touch; ②
        -ms-overflow-style: -ms-autohiding-scrollbar; ③
    }
    .table-responsive-md > .table-bordered {
        border: 0;
    }
}

@media (max-width: 991.98px) { ①
    .table-responsive-lg {
        display: block;
        width: 100%;
        overflow-x: auto;
        -webkit-overflow-scrolling: touch; ②
        -ms-overflow-style: -ms-autohiding-scrollbar; ③
    }
    .table-responsive-lg > .table-bordered {
        border: 0;
    }
}

@media (max-width: 1199.98px) { ①
    .table-responsive-xl {
        display: block;
        width: 100%;
        overflow-x: auto;
        -webkit-overflow-scrolling: touch; ②
        -ms-overflow-style: -ms-autohiding-scrollbar; ③
    }
    .table-responsive-xl > .table-bordered {
        border: 0;
    }
}
```

3

## 5

SECTION

## 図表

画像とキャプションとがセットになったような図表コンテンツを作成する場合、**figure** 要素を使ってマークアップを行います。

## 3.5.1

## 図表の基本的なスタイリング

Bootstrap では、**figure** 要素および **figcaption** 要素に **figure** クラス、**figure-img** クラス、**figure-caption** クラスを追加して、図表コンテンツを作成します。**figure** 要素内に含まれる画像をレスポンシブ対応させるためには、**img-fluid** クラスを子要素の **img** 要素に追加する必要があります（リスト 3-81、図 3-46）。

▼リスト 3-81 図表の基本的なスタイリング（figure-basic.html）

```
<figure class="figure">
  
  <figcaption class="figure-caption">画像についてのキャプション</figcaption>
</figure>
```

▼図 3-46 図表の基本的なスタイリング



なお、Bootstrap における **figure** 要素および **figure** クラスには、リスト 3-82 のようなスタイルで表示形式やマージンサイズ、文字設定が定義されています。

## ▼リスト3-82 figure要素、figureクラス、figure-imgクラス、figure-captionクラスに定義されているスタイル

```
article, aside, dialog, figcaption, figure, footer, header, hgroup, main, nav, section {
  display: block;
}
…中略…
figure {
  margin: 0 0 1rem;
}
…中略…
.figure {
  display: inline-block;
}

.figure-img {
  margin-bottom: 0.5rem;
  line-height: 1;
}

.figure-caption {
  font-size: 90%;
  color: #6c757d;
}
```

### 3.5.2 図表キャプションの位置合わせ

図表キャプションの位置合わせには、Textユーティリティ(P.347参照)の**text-left**クラス、**text-center**クラス、**text-right**クラスなどを使用できます(リスト3-83、図3-47)。

## ▼リスト3-83 図表キャプションの位置合わせ (figure-aligning.html)

```
<!-- キャプションの左揃え（デフォルト）-->
<div class="container">
  <figure class="figure">
    
    <figcaption class="figure-caption text-left">図表キャプション (text-left: デフォルト) </figcaption>
  </figure>
</div>

<!-- キャプションの中央揃え -->
<div class="container">
  <figure class="figure">
    
    <figcaption class="figure-caption text-center">図表キャプション (text-center) </figcaption>
  </figure>
</div>

<!-- キャプションの右寄せ -->
```

```
<div class="container">
  <figure class="figure">
    
    <figcaption class="figure-caption text-right">図表キャプション (text-right) </figcaption>
  </figure>
</div>
```

▼図 3-47 中央揃え（左）、右揃え（右）



## 3

SECTION

## 6

# Rebootによる初期設定

ここまで見てきたように、Bootstrapでは定義済みクラスを追加しない場合でも、ある程度体裁が整った状態で表示されますが、これは**Reboot**と呼ばれるしくみによって実現されています。**Reboot**(再起動の意味)は、**Normalize.css**を元に構築されたBootstrap特有の**リセットスタイル**です。Bootstrapにおけるブラウザやデバイス間の表示の不一致を修正し、クロスブラウザ対応のレンダリングを実現しています。

**Reboot**は、いくつかの基本要素のデフォルトスタイルに対し、タイプセレクタのみでリセットを行うことで、Bootstrapにおけるスタイル定義の基礎を作っています。

本節では、ユーティリティ(P.301参照)を使ってコンポーネントの配置を調整したり、独自CSSを追加してカスタマイズしたい人のために、BootstrapのRebootの内容について解説します。特に必要のない方は読み飛ばしてください。

## NOTE

## リセットCSSとノーマライズCSS

ブラウザが持っているデフォルトのスタイルは、ブラウザの種類やバージョンによって仕様が異なるため、Webページの表示に差異を生じることがあります。この問題を解決するために、各種ブラウザの持つデフォルトスタイルをいったんフラットな状態にした上でスタイルする**リセットCSS(Reset CSS)**という方法が生まれました。有名なものには、Eric MeyerのReset CSS(<https://meyerweb.com/eric/tools/css/reset/>)などがあります。リセットCSSとは、ほとんどの要素のマージンやパディングを0、行高さを1、フォントサイズを100%に指定し、ブラウザ表示の差異をリセットします。たとえば、見出し要素も段落要素もまったく同じ見た目になります。ただしこの方法には、ほとんどの要素のスタイルを再定義しなければならないというデメリットも生じます。

これに対し**ノーマライズCSS(Normalize CSS)**は、有用なデフォルトのスタイルは残しつつ、ブラウザ間の表示の差異を統一する方法です。

▼【公式サイト】Normalize.css: Make browsers render all elements more consistently.

<http://necolas.github.io/normalize.css/>

Bootstrapのリセット方法である**Reboot**は、この**ノーマライズCSS**をベースに構築されています。

### 3.6.1 Reboot によって初期化されているスタイル

**Reboot** では、基本的な要素のデフォルトスタイルがタイプセレクタのみで上書き（以降「リブート」）されます。リブートの基本設定は下記の方針に基づいています。

- コンポーネントのスペーシングを変更可能にするために、単位が **rem**（ルート要素の **font-size** の高さを 1 とする単位）に設定されている
- マージンの一方向性を重要視し、上方向のマージン：**margin-top** が排除されている
- あらゆるデバイスで拡大縮小を容易にするために、ブロック要素のマージンサイズの単位にも **rem** が設定されている
- フォント関連のプロパティの宣言を最小限に抑えるため、可能な限り継承を使う設定になっている

具体的なリブート設定については、次項以降で紹介していきます。

### 3.6.2 全要素へのリブート設定

まずは **html** 要素や **body** 要素などページ全体の初期設定のためのリブート設定を見ていきましょう。大前提となる要素の幅や高さの算出方法については、**box-sizing:border-box;** が指定され、要素の幅がパディングやボーダーのサイズに影響されないように設定されています（リスト 3-84）。

#### ▼リスト 3-84 全要素へのリブート設定

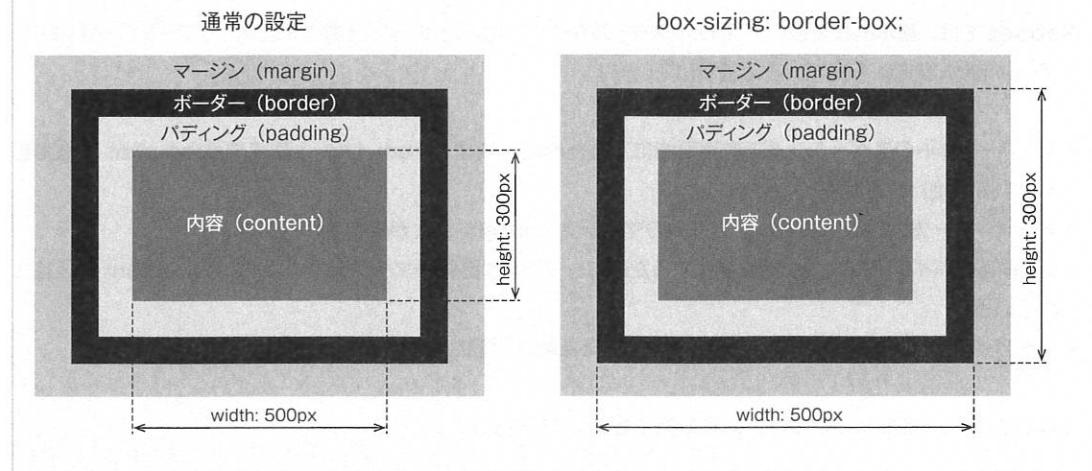
```
*,
*:before,
*:after {
  box-sizing: border-box; /* ボックスサイズの算出方法：パディングとボーダーを幅と高さに含める */
}
```



#### box-sizing プロパティ

**box-sizing** プロパティは、ボックスサイズの算出方法を指定するプロパティです。スタイルシートにおいて **box-sizing: border-box;** を宣言すると、ボックスの幅と高さの値に、ボーダーとパディングの値を含めて算出してくれます（通常は、内容の幅にボーダーとパディングのサイズを含みません）（図 3-48）。

▼図3-48 通常の設定（左）、box-sizing: border-box; を宣言した設定（右）



### 3.6.3 body要素へのリブート設定

ほとんどのブラウザのデフォルト値として、フォントサイズには **16px** が設定されています。これを元にbody要素へのリブート設定として **font-size: 1rem** を宣言し、メディアクエリによるレスポンシブな文字サイズ調整を可能にしています（リスト3-85）。加えて、フォントファミリーには、あらゆるデバイスとOS上で最適なテキストのレンダリングを行うためにネイティブフォントが設定されています（①）。行高さには **line-height: 1.5** を、文字揃えには **text-align: left** を宣言し、各要素の文字スタイルの不一致を防いでいます（②）。また、すべてのブラウザでのマージンを削除し、背景色をブラウザ初期設定の **#fff** に戻します（③）。

▼リスト3-85 body要素へのリブート設定

```
body {
  margin: 0;
  font-family: -apple-system, BlinkMacSystemFont, "Segoe UI", Roboto, "Helvetica Neue", Arial, sans-serif, "Apple Color Emoji", "Segoe UI Emoji", "Segoe UI Symbol"; ①
  font-size: 1rem;
  font-weight: 400;
  line-height: 1.5; ②
  color: #212529;
  text-align: left; ②
  background-color: #fff; ③
}
```

### 3.6.4 見出しと段落へのリブート設定

見出し要素（**h1～h6**）および段落要素（**p**）からは、上マージンが削除されます（リスト3-86 ①）。また、

見出し要素には下マージン **margin-bottom: 0.5rem** が、段落要素には下マージン **margin-bottom: 1rem** が追加されます（②）。これによりスペーシングが容易になります。

というのも、要素間を調整しようとマージンを設定しても意図したようにスペーシングできないケースがあります。たとえば、見出し要素に下マージン 2rem が設定され、後に続く段落要素に上マージン 1rem が設定されていた場合、合計した 3rem にはなりません。マージンの仕様上、値が大きい方に相殺されるので、この場合は 2rem 分のスペースが開きます。下マージンのみを使うというルールの元でコーディングすれば、要素間のスペースが、上の要素の下マージンなのか、下の要素の上マージンなのか、どちらが影響しているのかを考えなくて済み、スペーシングが容易になるというわけです。

#### ▼リスト 3-86 見しおおよび段落要素へのリブート設定

```
h1, h2, h3, h4, h5, h6 {
  margin-top: 0; _____①
  margin-bottom: 0.5rem; _____②
}
p {
  margin-top: 0; _____①
  margin-bottom: 1rem; _____②
}
```

### 3.6.5 リストへのリブート設定

**ul** 要素、**ol** 要素、および **dl** 要素のすべてのリストからは上マージンが削除されます（リスト 3-87 ①）。また下マージン **margin-bottom: 1rem** が追加されます（②）。ネストされたリストからは下マージンが削除されます（③）。ネストするたびに不自然な下マージンが追加されるようなことがないスタイルとなっています。

#### ▼リスト 3-87 リスト要素へのリブート設定

```
ol,
ul,
dl {
  margin-top: 0; _____①
  margin-bottom: 1rem; _____②
}
ol ol,
ul ul,
ol ul,
ul ol {
  margin-bottom: 0; _____③
}
```

また、**dt** 要素は太字で表示されます（リスト 3-88 ①）。**dd** 要素からは下マージン **margin-bottom: .5rem** が追加され、左マージンが削除されます（②③）。

これにより、dl要素のスタイルのシンプル化、ヒエラルキーの解消、スペーシングの改善を行います。

▼リスト3-88 dt、dd要素へのリブート設定

```
dt {  
    font-weight: 700; ——————①  
}  
dd {  
    margin-bottom: .5rem; ——————②  
    margin-left: 0; ——————③  
}
```

### 3.6.6 整形済みのテキストへのリブート設定

整形済みのテキストを表すpre要素からは上マージンが削除されます（リスト3-89）。また下マージンmargin-bottom: 1remが追加されます（②）。

▼リスト3-89 pre要素へのリブート設定

```
pre {  
    margin-top: 0; ——————①  
    margin-bottom: 1rem; ——————②  
    overflow: auto;  
    -ms-overflow-style: scrollbar;  
}
```

### 3.6.7 テーブルへのリブート設定

table要素は枠線が単一線表示に設定されます（リスト3-90①）。caption要素にはパディングサイズ（②）や文字色設定（③）、文字の左揃え（④）、キャプションの表示位置（⑤）が定義されています。またth要素では一貫したテキスト整列を確保するように親要素の値を継承します（⑥）。

また、ボーダー、パディングなどに追加変更を行う場合は、tableクラス（P.73参照）を使用します。

▼リスト3-90 テーブル要素へのリブート設定

```
table {  
    border-collapse: collapse; ——————①  
}  
  
caption {  
    padding-top: 0.75rem; ——————②  
    padding-bottom: 0.75rem; ——————③  
    color: #868e96; ——————④  
    text-align: left; ——————⑤  
}
```

```

    caption-side: bottom; -----⑤
}

th {
    text-align: inherit; -----⑥
}

```

## 3.6.8 フォームへのリブート設定

よりシンプルなスタイルを実現するために、さまざまなフォーム要素がリブートされます。

### fieldset 要素へのリブート

**fieldset** 要素のボーダー、パディング、マージンを削除し、個々の入力コントロールや入力グループをまとめ要素として簡単に使用できるように設定します（リスト 3-91）。

▼リスト 3-91 fieldset 要素へのリブート設定

```

fieldset {
    min-width: 0;
    padding: 0;
    margin: 0;
    border: 0;
}

```

### legend 要素へのリブート

**legend** 要素は、入力グループの見出しとして表示しやすいように要素をブロックボックスとして表示し（リスト 3-92）、幅とパディングサイズを調整します。また下マージンを 0.5rem 確保し（②）、文字サイズを 1.5rem に設定します（③）。行高さや文字色は親要素から継承し（④）、スペース、タブ、改行の表示方法を標準値に戻します（⑤）。

▼リスト 3-92 legend 要素へのリブート設定

```

legend {
    display: block; -----①
    width: 100%;
    max-width: 100%;
    padding: 0;
    margin-bottom: .5rem; -----②
    font-size: 1.5rem; -----③
    line-height: inherit; -----④
    color: inherit; -----④
    white-space: normal; -----⑤
}

```

## ■ **label** 要素へのリブート

**label** 要素は、表示形式を `display: inline-block` に設定し、下マージンが追加されます（リスト 3-93）。

### ▼リスト 3-93 **label** 要素へのリブート設定

```
label {
  display: inline-block;
  margin-bottom: 0.5rem;
}
```

## ■ **input** 要素、**select** 要素、**textarea** 要素、**button** 要素へのリブート

**input** 要素、**select** 要素、**textarea** 要素、**button** 要素は、マージンが削除され、行高さの継承が追加されます（リスト 3-94）。

また、Android 4（webkit）ブラウザで `audio` 要素と `video` 要素のコントロールのバグを避ける設定（①）や、Firefox（moz）ブラウザで、`button` 要素と `input` 要素の内側のパディングを削除する設定（②）、Mobile Safari（webkit）ブラウザで、テキストが入力内で垂直方向にセンタリングされないバグを避ける設定（③）は、ノーマライズ CSS の記述を引き継いでいます。

### ▼リスト 3-94 **input** 要素、**select** 要素、**textarea** 要素、**button** 要素へのリブート設定

```
input,
button,
select,
optgroup,
textarea {
  margin: 0;
  font-family: inherit;
  font-size: inherit;
  line-height: inherit;
}
button,
input {
  overflow: visible;
}
button,
select {
  text-transform: none;
}
button,
html [type="button"],
[type="reset"],
[type="submit"] {
  -webkit-appearance: button; ①
}
button::-moz-focus-inner,
```

```
[type="button"]::-moz-focus-inner,  
[type="reset"]::-moz-focus-inner,  
[type="submit"]::-moz-focus-inner {  
  padding: 0;  
  border-style: none;  
}  
  
input[type="radio"],  
input[type="checkbox"] {  
  box-sizing: border-box;  
  padding: 0;  
}  
  
input[type="date"],  
input[type="time"],  
input[type="datetime-local"],  
input[type="month"] {  
  -webkit-appearance: listbox; —③  
}
```

3

## ■ **textarea** 要素へのリブート

**textarea** 要素は、水平方向のサイズ変更がページレイアウトを崩すことがあるため、垂直方向のサイズだけを変更できるように設定されます（リスト 3-95）。

### ▼リスト 3-95 **textarea** 要素へのリブート設定

```
textarea {  
  overflow: auto;  
  resize: vertical;  
}
```

9

10

## 3.6.9 リンクへのリブート設定

**a** 要素については、ノーマライズ CSS で定義されているスタイルに加えて、下記の設定が追加されます。

### ■ 文字色

**a** 要素について、リンク部分（a）およびホバー時（a:hover）の基本色が追加されます（表 3-4）。

▼表 3-4 リンク色の設定

| リンク     | 色       |
|---------|---------|
| a       | #007bff |
| a:hover | #0056b3 |

## ■ 文字装飾

リンク部分のアンダーライン装飾は、ホバー時（a:hover）のみに制限されます。このうち、href 属性や tagindex 属性のないリンクについては、アンダーライン装飾がなしになります。また、href 属性や tagindex 属性のないリンクについては、フォーカス時「a:focus」のアウトラインが非表示に設定されます（リスト 3-96）。

また iOS 8 以降と Safari 8 以降のアンダーラインの隙間を削除するノーマライズ CSS の記述を引き継いでいます（❶）。

### ▼リスト 3-96 リンク要素へのリブート設定

```
a {
  color: #007bff;
  text-decoration: none;
  background-color: transparent;
  -webkit-text-decoration-skip: objects; ❶
}
a:hover {
  color: #0056b3;
  text-decoration: underline;
}
a:not([href]):not([tabindex]) {
  color: inherit;
  text-decoration: none;
}
a:not([href]):not([tabindex]):focus, a:not([href]):not([tabindex]):hover {
  color: inherit;
  text-decoration: none;
}
a:not([href]):not([tabindex]):focus {
  outline: 0;
}
```

## 3.6.10 その他要素へのリブート設定

他の要素の主なリブート設定を下記に挙げます。

### ■ address 要素へのリブート

**address** 要素は、下マージン margin-bottom: 1rem が追加されます。また、フォントスタイルがデフォルトのイタリックからノーマルにリセットされ、行高さの継承が追加されます（リスト 3-97）。

### ▼リスト 3-97 textarea 要素へのリブート設定

```
address {
  margin-bottom: 1rem;
  font-style: normal;
```

```
    line-height: inherit;
}
```

## blockquote 要素へのリブート

引用を表す **blockquote** 要素のデフォルトマージンは、上下 1em、左右 40px ですが、他の要素と一貫性を保つために、上マージンと左右マージンを削除し、下マージン **margin-bottom: 1rem** が追加されます（リスト 3-98）。

▼リスト 3-98 blockquote 要素へのリブート設定

```
blockquote {
  margin: 0 0 1rem;
```

## abbr 要素へのリブート

略称を表す **abbr** 要素は、段落テキストの中で目立つように、点線のアンダーラインなど基本的なスタイルに設定されます（リスト 3-99）。また、マウスポインターの表示を、クエスチョンマークの付いたポインターに設定します（➊）。

▼リスト 3-99 abbr 要素へのリブート設定

```
abbr[title],
abbr[data-original-title] {
  text-decoration: underline;
  -webkit-text-decoration: underline dotted;
  text-decoration: underline dotted;
  cursor: help; ❶
  border-bottom: 0;
}
```

### 3.6.11 HTML5 の hidden 属性

HTML5 では、**hidden** というグローバル属性が追加されました。**hidden** 属性のある要素には、デフォルトスタイルとして要素を非表示にするための **display:none** が設定されます。Bootstrap では、この要素に誤って **display** プロパティが再設定されないように **[hidden] { display: none !important; }** がリブート設定されます（リスト 3-100）。

この明示的な宣言によって、**hidden** 属性がサポートされない Internet Explorer 10 における問題を回避できるようになります。

▼リスト 3-100 hidden 属性のある要素へのリブート設定

```
[hidden] {
```

1  
2  
3  
4  
5  
6  
7  
8  
9  
10

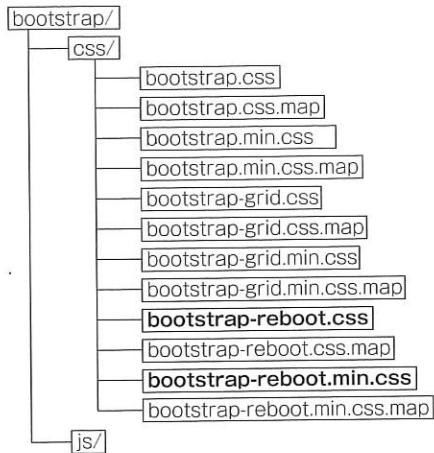
```
display: none !important;  
}
```

なお、要素の表示形式（display）を変更せずに、可視性（visibility）だけを切り替える場合には、**Visibility ユーティリティ**（P.354 参照）で定義された **invisible** クラスを使用します。

## COLUMN リセットスタイルとしての Reboot

「Bootstrap の導入」（P.14 参照）でも触れましたが、Bootstrap のダウンロードファイル内には、リブート設定単独の CSS ファイル 「bootstrap-reboot.css」 や、「bootstrap-reboot.min.css」 も用意されているため、Bootstrap 以外での一般的なリセットスタイルとして活用することも可能です。

### ▼ ダウンロードファイルのディレクトリ構成



また、そもそもリブート設定は、ソースコード版 Bootstrap 内の 「\_reboot.scss」 から CSS ファイルにコンパイルされるようになっています。「\_reboot.scss」 内には、各宣言やルールセットの意図が英語でコメント記述されていますので、詳しく知りたい方はご一読されると良いでしょう。ソースコード版の入手方法については「SCSS ファイルの準備」（P.443 参照）をご確認ください。