Layout

Contents

Forms

Every group of form fields should reside in a `<form>` element. Bootstrap provides no default styling for the `<form>` element, but there are some powerful browser features that are provided by default.

- New to browser forms? Consider reviewing the MDN form docs for an overview and complete list of available attributes.
- `<button>`s within a `<form>` default to `type="submit"`, so strive to be specific and always include a `type`.
- You can disable every form element within a form with the `disabled` attribute on the `<form>`.

Since Bootstrap applies `display: block` and `width: 100%` to almost all our form controls, forms will by default stack vertically. Additional classes can be used to vary this layout on a per-form basis.

Utilities

[Margin utilities]({{< docsref "/utilities/spacing" >}}) are the easiest way to add some structure to forms. They provide basic grouping of labels, controls, optional form text, and form validation messaging. We recommend sticking to `margin-bottom` utilities, and using a single direction throughout the form for consistency.

Feel free to build your forms however you like, with `<fieldset>`s, `<div>`s, or nearly any other element.

{{< example >}}

Example label

Another label

{{< /example >}}

Form grid

More complex forms can be built using our grid classes. Use these for form layouts that require multiple columns, varied widths, and additional alignment options. **Requires the `$enable-grid-classes` Sass variable to be enabled** (on by default).

{{< example >}}

```
<input type="text" class="form-control" placeholder="First name" aria-label="First name">
```

```
<input type="text" class="form-control" placeholder="Last name" aria-label="Last name">
```

{{< /example >}}

Gutters

By adding [gutter modifier classes]({{< docsref "/layout/gutters" >}}), you can have control over the gutter width in as well the inline as block direction. **Also requires the `$enable-grid-classes` Sass variable to be enabled** (on by default).

{{< example >}}

```
<input type="text" class="form-control" placeholder="First name" aria-label="First name">
```

```
<input type="text" class="form-control" placeholder="Last name" aria-label="Last name">
```

{{< /example >}}

More complex layouts can also be created with the grid system.

{{< example >}}

```
<label for="inputEmail4" class="form-label">Email</label>
<input type="email" class="form-control" id="inputEmail4">
```

```
<label for="inputPassword4" class="form-label">Password</label>
<input type="password" class="form-control" id="inputPassword4">
```

```
<label for="inputAddress" class="form-label">Address</label>
<input type="text" class="form-control" id="inputAddress" placeholder="1234 Main St">
```

```
<label for="inputAddress2" class="form-label">Address 2</label>
<input type="text" class="form-control" id="inputAddress2" placeholder="Apartment, studio,

<label for="inputCity" class="form-label">City</label>
<input type="text" class="form-control" id="inputCity">

<label for="inputState" class="form-label">State</label>
<select id="inputState" class="form-select">
  <option selected>Choose...</option>
  <option>...</option>
</select>

<label for="inputZip" class="form-label">Zip</label>
<input type="text" class="form-control" id="inputZip">

<div class="form-check">
  <input class="form-check-input" type="checkbox" id="gridCheck">
  <label class="form-check-label" for="gridCheck">
    Check me out
  </label>
</div>

<button type="submit" class="btn btn-primary">Sign in</button>
```

{{< /example >}}

Horizontal form

Create horizontal forms with the grid by adding the `.row` class to form groups
and using the `.col-*-*` classes to specify the width of your labels and controls.
Be sure to add `.col-form-label` to your `<label>`s as well so they're vertically
centered with their associated form controls.

At times, you maybe need to use margin or padding utilities to create that
perfect alignment you need. For example, we've removed the `padding-top` on
our stacked radio inputs label to better align the text baseline.

{{< example >}}

```
<label for="inputEmail3" class="col-sm-2 col-form-label">Email</label>
<div class="col-sm-10">
  <input type="email" class="form-control" id="inputEmail3">
</div>

<label for="inputPassword3" class="col-sm-2 col-form-label">Password</label>
<div class="col-sm-10">
  <input type="password" class="form-control" id="inputPassword3">
</div>
```

Radios

```
    <div class="form-check">
      <input class="form-check-input" type="radio" name="gridRadios" id="gridRadios1" value="c
      <label class="form-check-label" for="gridRadios1">
        First radio
      </label>
    </div>
    <div class="form-check">
      <input class="form-check-input" type="radio" name="gridRadios" id="gridRadios2" value="c
      <label class="form-check-label" for="gridRadios2">
        Second radio
      </label>
    </div>
    <div class="form-check disabled">
      <input class="form-check-input" type="radio" name="gridRadios" id="gridRadios3" value="c
      <label class="form-check-label" for="gridRadios3">
        Third disabled radio
      </label>
    </div>
</div>

<div class="col-sm-10 offset-sm-2">
  <div class="form-check">
    <input class="form-check-input" type="checkbox" id="gridCheck1">
    <label class="form-check-label" for="gridCheck1">
      Example checkbox
    </label>
  </div>
</div>
```

Sign in

{{< /example >}}

Horizontal form label sizing

Be sure to use `.col-form-label-sm` or `.col-form-label-lg` to your
`<label>`s or `<legend>`s to correctly follow the size of `.form-control-lg` and
`.form-control-sm`.

{{< example >}}

Email

```
<input type="email" class="form-control form-control-sm" id="colFormLabelSm" placeholder="co
```

Email

```
<input type="email" class="form-control" id="colFormLabel" placeholder="col-form-label">
```

Email

```
<input type="email" class="form-control form-control-lg" id="colFormLabelLg" placeholder="co
```

{{< /example >}}

Column sizing

As shown in the previous examples, our grid system allows you to place any number of `.cols` within a `.row`. They'll split the available width equally between them. You may also pick a subset of your columns to take up more or less space, while the remaining `.cols` equally split the rest, with specific column classes like `.col-sm-7`.

{{< example >}}

```
<input type="text" class="form-control" placeholder="City" aria-label="City">

<input type="text" class="form-control" placeholder="State" aria-label="State">

<input type="text" class="form-control" placeholder="Zip" aria-label="Zip">
```

{{< /example >}}

Auto-sizing

The example below uses a flexbox utility to vertically center the contents and changes `.col` to `.col-auto` so that your columns only take up as much space as needed. Put another way, the column sizes itself based on the contents.

{{< example >}}

```
<label class="visually-hidden" for="autoSizingInput">Name</label>
<input type="text" class="form-control" id="autoSizingInput" placeholder="Jane Doe">

<label class="visually-hidden" for="autoSizingInputGroup">Username</label>
<div class="input-group">
  <div class="input-group-text">@</div>
  <input type="text" class="form-control" id="autoSizingInputGroup" placeholder="Username">
</div>

<label class="visually-hidden" for="autoSizingSelect">Preference</label>
<select class="form-select" id="autoSizingSelect">
  <option selected>Choose...</option>
  <option value="1">One</option>
  <option value="2">Two</option>
  <option value="3">Three</option>
</select>

<div class="form-check">
  <input class="form-check-input" type="checkbox" id="autoSizingCheck">
  <label class="form-check-label" for="autoSizingCheck">
    Remember me
```

```
    </label>
  </div>

  <button type="submit" class="btn btn-primary">Submit</button>
```

{{< /example >}}

You can then remix that once again with size-specific column classes.

{{< example >}}

```
  <label class="visually-hidden" for="specificSizeInputName">Name</label>
  <input type="text" class="form-control" id="specificSizeInputName" placeholder="Jane Doe">

  <label class="visually-hidden" for="specificSizeInputGroupUsername">Username</label>
  <div class="input-group">
    <div class="input-group-text">@</div>
    <input type="text" class="form-control" id="specificSizeInputGroupUsername" placeholder="U
  </div>

  <label class="visually-hidden" for="specificSizeSelect">Preference</label>
  <select class="form-select" id="specificSizeSelect">
    <option selected>Choose...</option>
    <option value="1">One</option>
    <option value="2">Two</option>
    <option value="3">Three</option>
  </select>

  <div class="form-check">
    <input class="form-check-input" type="checkbox" id="autoSizingCheck2">
    <label class="form-check-label" for="autoSizingCheck2">
      Remember me
    </label>
  </div>

  <button type="submit" class="btn btn-primary">Submit</button>
```

{{< /example >}}

Inline forms

Use the `.row-cols-*` classes to create responsive horizontal layouts. By adding
[gutter modifier classes]({{< docsref "/layout/gutters" >}}), we'll have gutters
in horizontal and vertical directions. On narrow mobile viewports, the `.col-12`
helps stack the form controls and more. The `.align-items-center` aligns the
form elements to the middle, making the `.form-checkbox` align properly.

{{< example >}}

```
  <label class="visually-hidden" for="inlineFormInputGroupUsername">Username</label>
  <div class="input-group">
    <div class="input-group-text">@</div>
```

```
    <input type="text" class="form-control" id="inlineFormInputGroupUsername" placeholder="Use
</div>

<label class="visually-hidden" for="inlineFormSelectPref">Preference</label>
<select class="form-select" id="inlineFormSelectPref">
  <option selected>Choose...</option>
  <option value="1">One</option>
  <option value="2">Two</option>
  <option value="3">Three</option>
</select>

<div class="form-check">
  <input class="form-check-input" type="checkbox" id="inlineFormCheck">
  <label class="form-check-label" for="inlineFormCheck">
    Remember me
  </label>
</div>

<button type="submit" class="btn btn-primary">Submit</button>
```

$\{\{< /example >\}\}$