skillcrush

RAILS FORM HELPERS cheatsheet

COMMON FORM FIELDS

button_tag(content_or_options = nil, options = nil, &block)

Creates a button element that defines a submit button, reset button or a generic button which can be used in JavaScript, for example. You can use the button tag as a regular submit tag but it isn't supported in legacy browsers. However, the button tag allows richer labels such as images and emphasis, so this helper will also accept a block.

Options

- :data - This option can be used to add custom data attributes.
- :disabled - If true, the user will not be able to use this input.
- Any other key creates standard HTML options for the tag.

Data attributes

- confirm: 'question?' - If present, the unobtrusive JavaScript drivers will provide a prompt with the question specified. If the user accepts, the form is processed normally, otherwise no action is taken.
- :disable_with - Value of this parameter will be used as the value for a disabled version of the submit button when the form is submitted. This feature is provided by the unobtrusive JavaScript driver.

```
<%= f.button_tag %>
# => <button name="button" type="submit">Button</button>

<% f.button_tag(type: 'button') do %>
  <%= content_tag(:strong, 'Ask me!')%>
<% end %>
```

```
# => <button name="button" type="button">
#     <strong>Ask me!</strong>
#    </button>

<%= button_tag "Checkout", data: { disable_with: "Please wait..." } %>
# => <button data-disable-with="Please wait..." name="button"
type="submit">Checkout</button>
```

check_box_tag(name, value = "1", checked = false, options = {})

Creates a check box form input tag.

Options

- :disabled - If set to true, the user will not be able to use this input.

```
check_box_tag 'accept'
# => <input id="accept" name="accept" type="checkbox" value="1" />

check_box_tag 'rock', 'rock music'
# => <input id="rock" name="rock" type="checkbox" value="rock music" />

check_box_tag 'receive_email', 'yes', true
# => <input checked="checked" id="receive_email" name="receive_email"
type="checkbox" value="yes" />

check_box_tag 'tos', 'yes', false, class: 'accept_tos'
# => <input class="accept_tos" id="tos" name="tos" type="checkbox"
value="yes" />
check_box_tag 'eula', 'accepted', false, disabled: true
# => <input disabled="disabled" id="eula" name="eula" type="checkbox"
value="accepted" />
```

date_field_tag(name, value = nil, options = {})

Creates a text field of type “date”.

Options

- Accepts the same options as text_field_tag.

```
date_field_tag 'name'
# => <input id="name" name="name" type="date" />

date_field_tag 'date', '01/01/2014'
# => <input id="date" name="date" type="date" value="01/01/2014" />

date_field_tag 'date', nil, class: 'special_input'
# => <input class="special_input" id="date" name="date" type="date" />

date_field_tag 'date', '01/01/2014', class: 'special_input', disabled:
true
# => <input disabled="disabled" class="special_input" id="date"
name="date" type="date" value="01/01/2014" />
```

email_field_tag(name, value = nil, options = {})

Creates a text field of type “email”.

Options

- Accepts the same options as text_field_tag.

```
email_field_tag 'name'
# => <input id="name" name="name" type="email" />
email_field_tag 'email', 'email@example.com'
# => <input id="email" name="email" type="email"
value="email@example.com" />
email_field_tag 'email', nil, class: 'special_input'
# => <input class="special_input" id="email" name="email" type="email" />
```

```
email_field_tag 'email', 'email@example.com', class: 'special_input',
disabled: true
# => <input disabled="disabled" class="special_input" id="email"
name="email" type="email" value="email@example.com" />
```

file_field_tag(name, options = {}) Link

Creates a file upload field. If you are using file uploads then you will also need to set the multipart option for the form tag:

```
<%= form_tag '/upload', multipart: true do %>
  <label for="file">File to Upload</label> <%= file_field_tag "file" %>
  <%= submit_tag %>
<% end %>
```

The specified URL will then be passed a File object containing the selected file, or if the field was left blank, a StringIO object.

Options

- Creates standard HTML attributes for the tag.
- :disabled - If set to true, the user will not be able to use this input.
- :multiple - If set to true, *in most updated browsers* the user will be allowed to select multiple files.
- :accept - If set to one or multiple mime-types, the user will be suggested a filter when choosing a file. You still need to set up model validations.

```
file_field_tag 'attachment'
# => <input id="attachment" name="attachment" type="file" />
file_field_tag 'avatar', class: 'profile_input'
# => <input class="profile_input" id="avatar" name="avatar" type="file"
/>
```

```
file_field_tag 'picture', disabled: true
# => <input disabled="disabled" id="picture" name="picture" type="file"
/>
file_field_tag 'resume', value: '~/resume.doc'
# => <input id="resume" name="resume" type="file" value="~/resume.doc" />
file_field_tag 'user_pic', accept: 'image/png,image/gif,image/jpeg'
# => <input accept="image/png,image/gif,image/jpeg" id="user_pic"
name="user_pic" type="file" />
file_field_tag 'file', accept: 'text/html', class: 'upload', value:
'index.html'
# => <input accept="text/html" class="upload" id="file" name="file"
type="file" value="index.html" />
```

hidden_field_tag(name, value = nil, options = {})

Creates a hidden form input field used to transmit data that would be lost due to HTTP's statelessness or data that should be hidden from the user.

Options

- Creates standard HTML attributes for the tag.

```
hidden_field_tag 'tags_list'
# => <input id="tags_list" name="tags_list" type="hidden" />
hidden_field_tag 'token', 'VUBJKB23UIVI1UU1VOBVI@'
# => <input id="token" name="token" type="hidden"
value="VUBJKB23UIVI1UU1VOBVI@" />
hidden_field_tag 'collected_input', '', onchange: "alert('Input
collected!')"
# => <input id="collected_input" name="collected_input"
onchange="alert('Input collected!')"
#    type="hidden" value="" />
```

label_tag(name = nil, content_or_options = nil, options = nil, &block)

Creates a label element. Accepts a block.

Options

- Creates standard HTML attributes for the tag.

```
label_tag 'name'
# => <label for="name">Name</label>
label_tag 'name', 'Your name'
# => <label for="name">Your name</label>
label_tag 'name', nil, class: 'small_label'
# => <label for="name" class="small_label">Name</label>
```

number_field_tag(name, value = nil, options = {})

Creates a number field.

Options

- :min - The minimum acceptable value.
- :max - The maximum acceptable value.
- :in - A range specifying the :min and :max values.
- :within - Same as :in.
- :step - The acceptable value granularity.
- Otherwise accepts the same options as text_field_tag.

```
number_field_tag 'quantity'
# => <input id="quantity" name="quantity" type="number" />
number_field_tag 'quantity', '1'
# => <input id="quantity" name="quantity" type="number" value="1" />
number_field_tag 'quantity', nil, class: 'special_input'
# => <input class="special_input" id="quantity" name="quantity"
type="number" />
number_field_tag 'quantity', nil, min: 1
# => <input id="quantity" name="quantity" min="1" type="number" />
number_field_tag 'quantity', nil, max: 9
```

```
# => <input id="quantity" name="quantity" max="9" type="number" />
number_field_tag 'quantity', nil, in: 1...10
# => <input id="quantity" name="quantity" min="1" max="9" type="number"
/>
number_field_tag 'quantity', nil, within: 1...10
# => <input id="quantity" name="quantity" min="1" max="9" type="number"
/>
number_field_tag 'quantity', nil, min: 1, max: 10
# => <input id="quantity" name="quantity" min="1" max="9" type="number"
/>
number_field_tag 'quantity', nil, min: 1, max: 10, step: 2
# => <input id="quantity" name="quantity" min="1" max="9" step="2"
type="number" />
number_field_tag 'quantity', '1', class: 'special_input', disabled: true
# => <input disabled="disabled" class="special_input" id="quantity"
name="quantity" type="number" value="1" />
```

password_field_tag(name = "password", value = nil, options = {})

Creates a password field, a masked text field that will hide the users input behind a mask character.

Options

- :disabled - If set to true, the user will not be able to use this input.
- :size - The number of visible characters that will fit in the input.
- :maxlength - The maximum number of characters that the browser will allow the user to enter.
- Any other key creates standard HTML attributes for the tag.

```
password_field_tag 'pass'
# => <input id="pass" name="pass" type="password" />
password_field_tag 'secret', 'Your secret here'
```

```
# => <input id="secret" name="secret" type="password" value="Your secret
here" />
password_field_tag 'masked', nil, class: 'masked_input_field'
# => <input class="masked_input_field" id="masked" name="masked"
type="password" />
password_field_tag 'token', '', size: 15
# => <input id="token" name="token" size="15" type="password" value="" />
password_field_tag 'key', nil, maxlength: 16
# => <input id="key" maxlength="16" name="key" type="password" />
password_field_tag 'confirm_pass', nil, disabled: true
# => <input disabled="disabled" id="confirm_pass" name="confirm_pass"
type="password" />
password_field_tag 'pin', '1234', maxlength: 4, size: 6, class:
"pin_input"
# => <input class="pin_input" id="pin" maxlength="4" name="pin" size="6"
type="password" value="1234" />
```

radio_button_tag(name, value, checked = false, options = {})

Creates a radio button; use groups of radio buttons named the same to allow users to select from a group of options.

Options

- :disabled - If set to true, the user will not be able to use this input.
- Any other key creates standard HTML options for the tag.

```
radio_button_tag 'gender', 'male'
# => <input id="gender_male" name="gender" type="radio" value="male" />

radio_button_tag 'receive_updates', 'no', true
# => <input checked="checked" id="receive_updates_no"
name="receive_updates" type="radio" value="no" />

radio_button_tag 'time_slot', "3:00 p.m.", false, disabled: true
```

```
# => <input disabled="disabled" id="time_slot_300_pm" name="time_slot"
type="radio" value="3:00 p.m." />

radio_button_tag 'color', "green", true, class: "color_input"
# => <input checked="checked" class="color_input" id="color_green"
name="color" type="radio" value="green" />
```

search_field_tag(name, value = nil, options = {})

Creates a text field of type "search".

Options

- Accepts the same options as text_field_tag.

```
search_field_tag 'name'
# => <input id="name" name="name" type="search" />

search_field_tag 'search', 'Enter your search query here'
# => <input id="search" name="search" type="search" value="Enter your
search query here" />

search_field_tag 'search', nil, class: 'special_input'
# => <input class="special_input" id="search" name="search" type="search"
/>

search_field_tag 'search', 'Enter your search query here', class:
'special_input', disabled: true
# => <input disabled="disabled" class="special_input" id="search"
name="search" type="search" value="Enter your search query here" />
```

select_tag(name, option_tags = nil, options = {})

Creates a dropdown selection box, or if the :multiple option is set to true, a multiple choice selection box.

Helpers::FormOptions can be used to create common select boxes such as countries, time zones, or associated records. option_tags is a string containing the option tags for the select box.

Options

- :multiple - If set to true the selection will allow multiple choices.
- :disabled - If set to true, the user will not be able to use this input.
- :include_blank - If set to true, an empty option will be created. If set to a string, the string will be used as the option's content and the value will be empty.
- :prompt - Create a prompt option with blank value and the text asking user to select something.
- Any other key creates standard HTML attributes for the tag.

```
select_tag "people", options_from_collection_for_select(@people, "id",
"name")
# <select id="people" name="people"><option
value="1">David</option></select>

select_tag "people", options_from_collection_for_select(@people, "id",
"name", "1")
# <select id="people" name="people"><option value="1"
selected="selected">David</option></select>

select_tag "people", "<option>David</option>".html_safe
# => <select id="people" name="people"><option>David</option></select>

select_tag "count",
"<option>1</option><option>2</option><option>3</option><option>4</option>
".html_safe
# => <select id="count" name="count"><option>1</option><option>2</option>
#    <option>3</option><option>4</option></select>
```

```
select_tag "colors",
"<option>Red</option><option>Green</option><option>Blue</option>".html_sa
fe, multiple: true
# => <select id="colors" multiple="multiple"
name="colors[]"><option>Red</option>
#    <option>Green</option><option>Blue</option></select>
select_tag "locations", "<option>Home</option><option
selected='selected'>Work</option><option>Out</option>".html_safe
# => <select id="locations" name="locations"><option>Home</option><option
selected='selected'>Work</option>
#    <option>Out</option></select>
select_tag "access",
"<option>Read</option><option>Write</option>".html_safe, multiple: true,
class: 'form_input', id: 'unique_id'
# => <select class="form_input" id="unique_id" multiple="multiple"
name="access[]"><option>Read</option>
#    <option>Write</option></select>
select_tag "people", options_from_collection_for_select(@people, "id",
"name"), include_blank: true
# => <select id="people" name="people"><option value=""></option><option
value="1">David</option></select>
select_tag "people", options_from_collection_for_select(@people, "id",
"name"), include_blank: "All"
# => <select id="people" name="people"><option
value="">All</option><option value="1">David</option></select>
select_tag "people", options_from_collection_for_select(@people, "id",
"name"), prompt: "Select something"
# => <select id="people" name="people"><option value="">Select
something</option><option value="1">David</option></select>
select_tag "destination",
"<option>NYC</option><option>Paris</option><option>Rome</option>".html_sa
fe, disabled: true
# => <select disabled="disabled" id="destination"
name="destination"><option>NYC</option>
```

```
#    <option>Paris</option><option>Rome</option></select>
select_tag "credit_card", options_for_select([ "VISA", "MasterCard" ],
"MasterCard")
# => <select id="credit_card" name="credit_card"><option>VISA</option>
#    <option selected="selected">MasterCard</option></select>
```

submit_tag(value = "Save changes", options = {})

Creates a submit button with the text value as the caption.

Options

- :data - This option can be used to add custom data attributes.
- :disabled - If true, the user will not be able to use this input.
- Any other key creates standard HTML options for the tag.

Data attributes

- confirm: 'question?' - If present the unobtrusive JavaScript drivers will provide a prompt with the question specified. If the user accepts, the form is processed normally, otherwise no action is taken.
- :disable_with - Value of this parameter will be used as the value for a disabled version of the submit button when the form is submitted. This feature is provided by the unobtrusive JavaScript driver.

```
submit_tag
# => <input name="commit" type="submit" value="Save changes" />

submit_tag "Edit this article"
# => <input name="commit" type="submit" value="Edit this article" />

submit_tag "Save edits", disabled: true
# => <input disabled="disabled" name="commit" type="submit" value="Save
edits" />
```

```
submit_tag "Complete sale", data: { disable_with: "Please wait..." }
# => <input name="commit" data-disable-with="Please wait..."
type="submit" value="Complete sale" />

submit_tag nil, class: "form_submit"
# => <input class="form_submit" name="commit" type="submit" />

submit_tag "Edit", class: "edit_button"
# => <input class="edit_button" name="commit" type="submit" value="Edit"
/>

submit_tag "Save", data: { confirm: "Are you sure?" }
# => <input name='commit' type='submit' value='Save' data-confirm="Are
you sure?" />
```

text_area_tag(name, content = nil, options = {})

Creates a text input area; use a textarea for longer text inputs such as blog posts or descriptions.

Options

- :size - A string specifying the dimensions (columns by rows) of the textarea (e.g., “25x10”).
- :rows - Specify the number of rows in the textarea
- :cols - Specify the number of columns in the textarea
- :disabled - If set to true, the user will not be able to use this input.
- :escape - By default, the contents of the text input are HTML escaped. If you need unescaped contents, set this to false.
- Any other key creates standard HTML attributes for the tag.

```
text_area_tag 'post'
# => <textarea id="post" name="post"></textarea>

text_area_tag 'bio', @user.bio
```

```
# => <textarea id="bio" name="bio">This is my biography.</textarea>
text_area_tag 'body', nil, rows: 10, cols: 25
# => <textarea cols="25" id="body" name="body" rows="10"></textarea>

text_area_tag 'body', nil, size: "25x10"
# => <textarea name="body" id="body" cols="25" rows="10"></textarea>

text_area_tag 'description', "Description goes here.", disabled: true
# => <textarea disabled="disabled" id="description"
name="description">Description goes here.</textarea>

text_area_tag 'comment', nil, class: 'comment_input'
# => <textarea class="comment_input" id="comment"
name="comment"></textarea>
```

text_field_tag(name, value = nil, options = {})

Creates a standard text field; use these text fields to input smaller chunks of text like a username or a search query.

Options

- :disabled - If set to true, the user will not be able to use this input.
- :size - The number of visible characters that will fit in the input.
- :maxlength - The maximum number of characters that the browser will allow the user to enter.
- :placeholder - The text contained in the field by default which is removed when the field receives focus.
- Any other key creates standard HTML attributes for the tag.

```
text_field_tag 'name'
# => <input id="name" name="name" type="text" />

text_field_tag 'query', 'Enter your search query here'
```

```
# => <input id="query" name="query" type="text" value="Enter your search
query here" />

text_field_tag 'search', nil, placeholder: 'Enter search term...'
# => <input id="search" name="search" placeholder="Enter search term..."
type="text" />

text_field_tag 'request', nil, class: 'special_input'
# => <input class="special_input" id="request" name="request" type="text"
/>

text_field_tag 'address', '', size: 75
# => <input id="address" name="address" size="75" type="text" value="" />

text_field_tag 'zip', nil, maxlength: 5
# => <input id="zip" maxlength="5" name="zip" type="text" />

text_field_tag 'payment_amount', '$0.00', disabled: true
# => <input disabled="disabled" id="payment_amount" name="payment_amount"
type="text" value="$0.00" />

text_field_tag 'ip', '0.0.0.0', maxlength: 15, size: 20, class:
"ip-input"
# => <input class="ip-input" id="ip" maxlength="15" name="ip" size="20"
type="text" value="0.0.0.0" />
```

url_field_tag(name, value = nil, options = {}) Brand

Creates a text field of type "url".

Options

- Accepts the same options as text_field_tag.

```
url_field_tag 'name'
```

```
# => <input id="name" name="name" type="url" />

url_field_tag 'url', 'http://rubyonrails.org'
# => <input id="url" name="url" type="url" value="http://rubyonrails.org"
/>

url_field_tag 'url', nil, class: 'special_input'
# => <input class="special_input" id="url" name="url" type="url" />

url_field_tag 'url', 'http://rubyonrails.org', class: 'special_input',
disabled: true
# => <input disabled="disabled" class="special_input" id="url" name="url"
type="url" value="http://rubyonrails.org" />
```