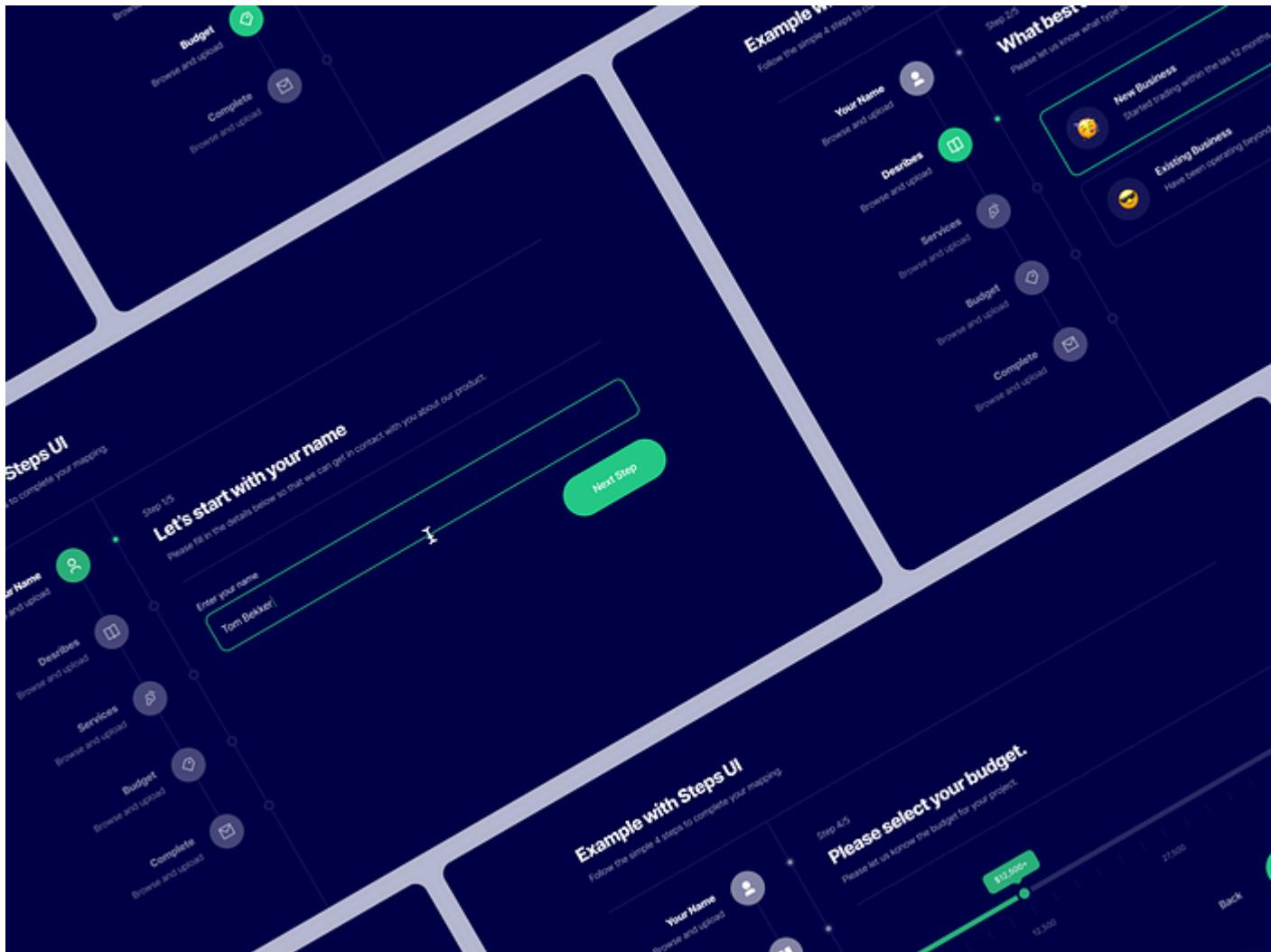


[< Go to the original](#)

Making forms more user-friendly

Practical tips for designing simple and inclusive forms for the web.

**Omar Andani**[Follow](#)

UX Collective a11y-light ~10 min read · June 20, 2021 (Updated: January 6, 2022) · Free: No

I was recently working on creating a better form experience for a project at work when it dawned on me that I had no idea what separated a good form from a bad one. Do I design a multi-step or a single-page form? Should I use placeholder text or not? How do I make my forms more inclusive?

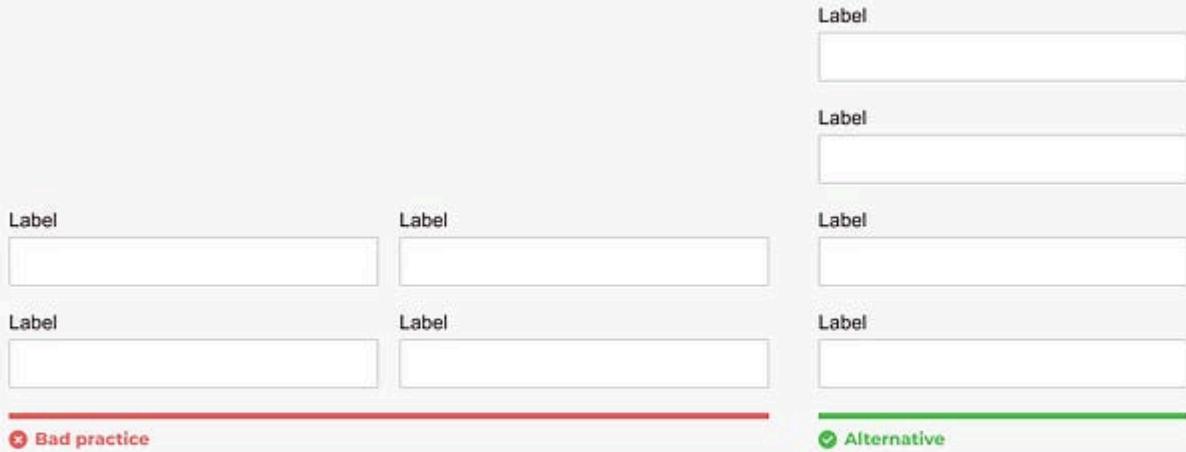
is achieved via forms. So while they may not be the most exciting element to design, good form design is an extremely valuable skill to possess.

After going through a few different resources I was able to compile a shortlist of the best practices that I found the most effective. However, keep in mind that forms are highly contextual elements — and exceptions to best practices are often required.

Before we begin, let's quickly cover the five main components of a form:

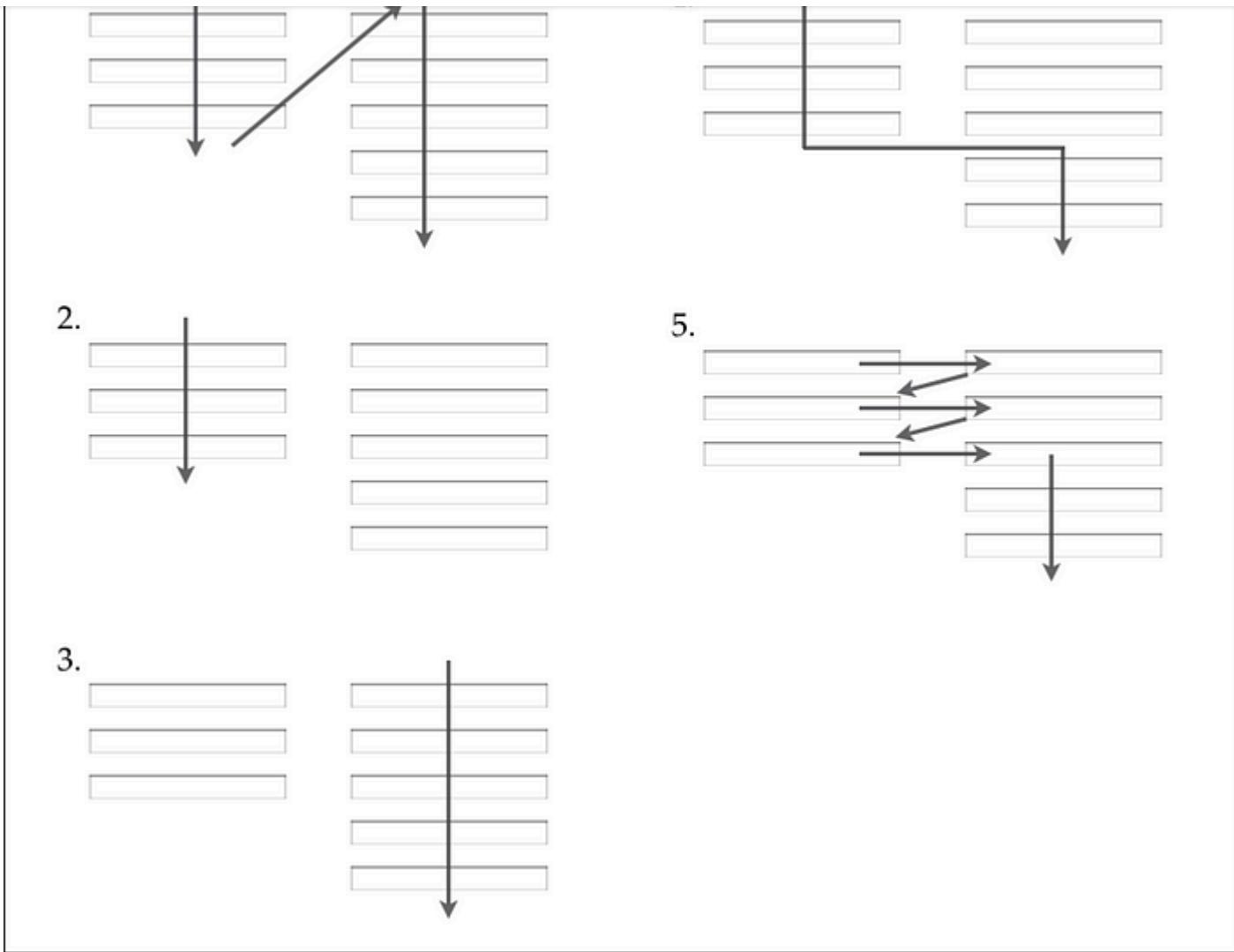
- **Structure** — the layout of the form and the order of the individual fields.
- **Input fields** — interface elements such as text boxes, radio buttons or checkboxes.
- **Field labels** — Descriptions for the corresponding input field.
- **Action button** — Allow the user to perform an action such as submitting their information.
- **Feedback** — Allows users to determine whether their submission was successful or not. Feedback can be given with visual cues or auditory ones.

1. Keep it one column



One-column forms are more comfortable to scan. [Source](#)

Multi-column form layouts are prone to misinterpretation — users are more likely to skip required fields, input data into the wrong fields or simply abandon the form entirely. The image below illustrates how users tend to misinterpret multi-column forms:



Some users may think they only have to fill out one column. Some may believe that there's a sequence they have to follow. Using multiple columns exposes the form to misinterpretation.

When a form offers no clear path to completion it takes much longer to comprehend it, let alone complete it. When the path to completion is linear (i.e. one column), there is no ambiguity as to how a form should be completed.

2. Flow, order, and grouping

Remove all non-essential fields

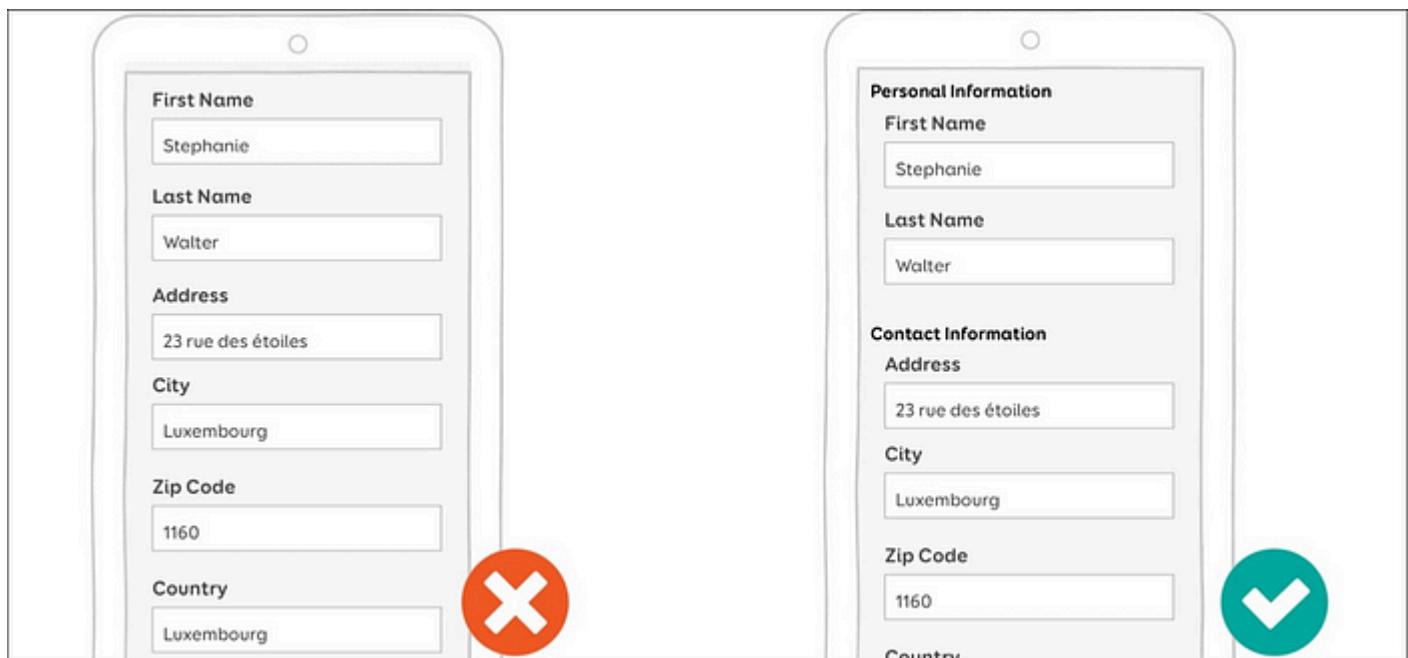
Make sure to only ask for information that you absolutely need. There could be \$12M on the line. Remember, the more fields you add the less likely people are to fill them all out, which can then affect things like conversion rate and result in loss of sales. Always ask and understand why you need certain pieces of information. One way to determine necessary form fields is to use the question

Order the form logically

It's all about asking for the right information at the right time — otherwise it can be a major turn-off for users. Think about your own online buying experiences; how would you feel if you were asked to provide your payment details before having an opportunity to review your order? Or having to fill in your address before you've had a chance to fill in your name? Try to envision form-building from the users' perspective.

Group related fields

Creating logical groups allows the user to process the information with more clarity, since they can focus on one group at a time rather than being overwhelmed with too many requests. Grouping information signifies association. It's also important to visually separate groups with added padding to distinguish them.



Grouping together related fields makes it easier for the user to process the information. [Source](#)

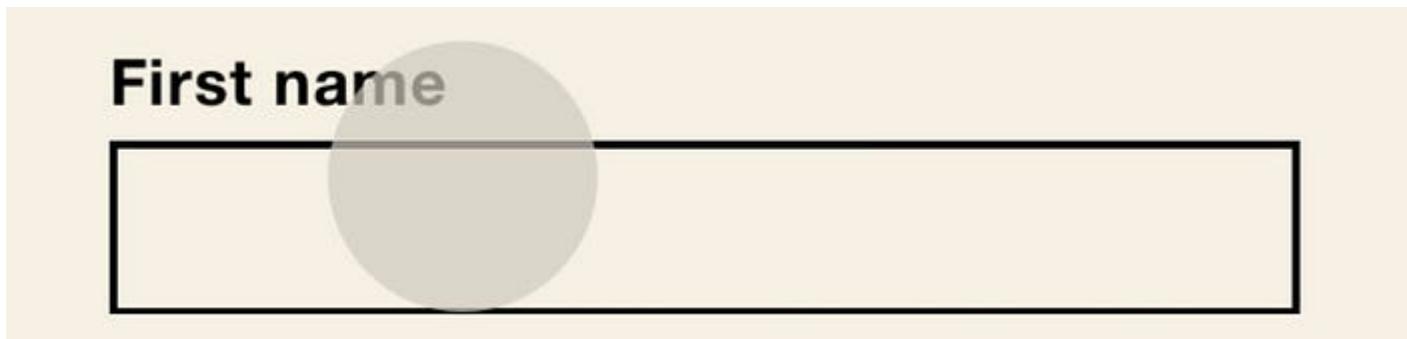
3. Use clear labels

To start, labels should be top-aligned where possible. According to [eye-tracking studies](#), putting labels above input fields — rather than adjacent to them — results

The image shows the 'Create account' form from the Amazon website. At the top is the Amazon logo. Below it is a large rectangular input field labeled 'Your name'. Underneath are fields for 'Email' and 'Password', each with a placeholder message ('at least 6 characters'). Below the password field is another for 'Password again'. At the bottom of the form is a yellow button labeled 'Create your Amazon account'. Below the button is a small note about agreeing to the 'Conditions of Use' and 'Privacy Notice'. At the very bottom is a link for users who already have an account: 'Already have an account? Sign in >'.

Amazon using top-aligned labels on their create an account screen.

Labels are also crucial from an accessibility standpoint. Sighted users can read them, visually-impaired users can hear them with help from a screen reader, and motor-impaired users can interact with the field more easily thanks to the larger hit area.

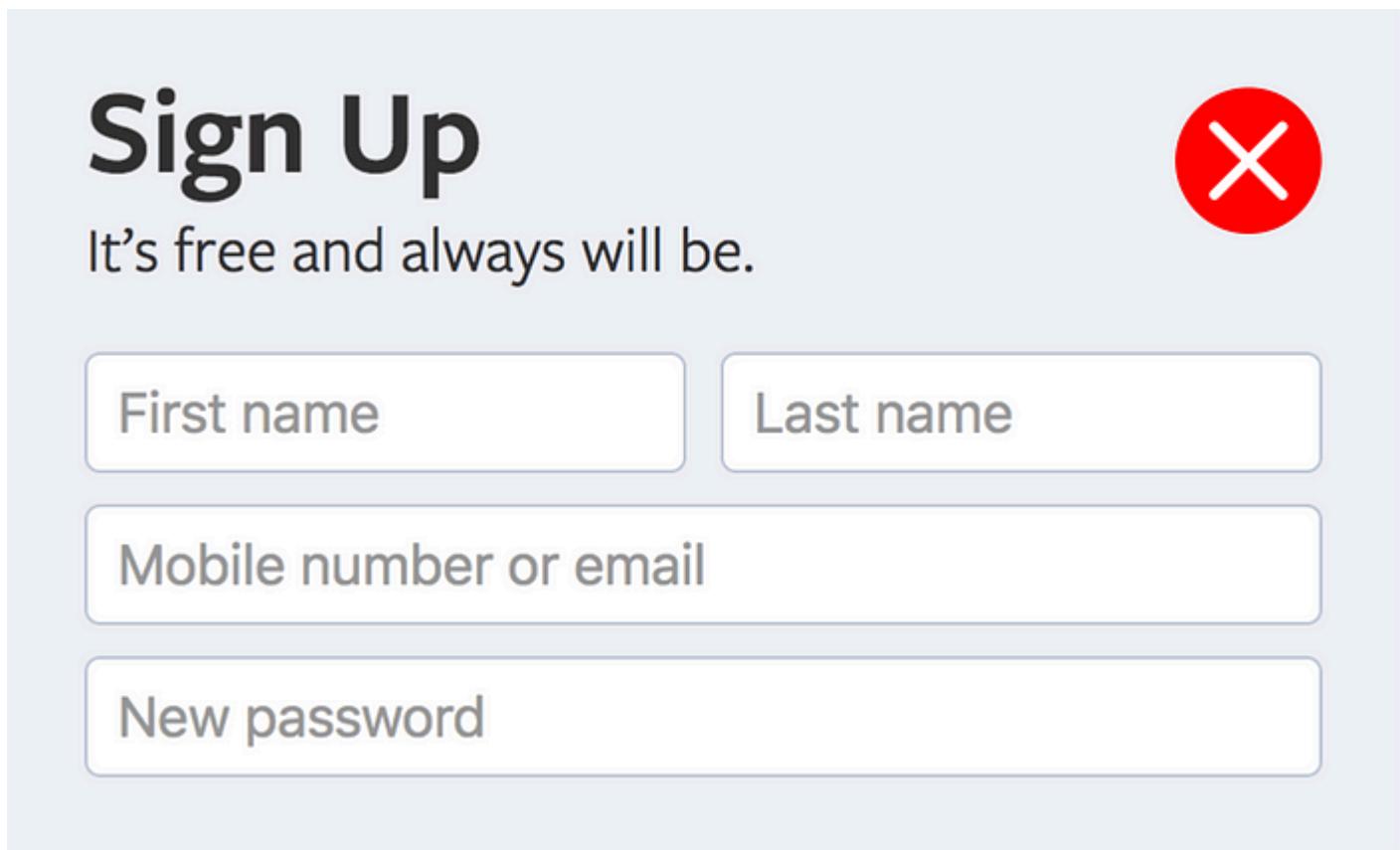


Using a top-aligned label increases the touch target size. Photo from Adam Silver's book: [Form Design Patterns](#).

4. Avoid placeholder text

Just because the *placeholder* attribute exists (in HTML specification) doesn't mean you have to use it. The reason for placeholders is that they provide users with extra guidance when filling out a field. But the problem is that they are greyed

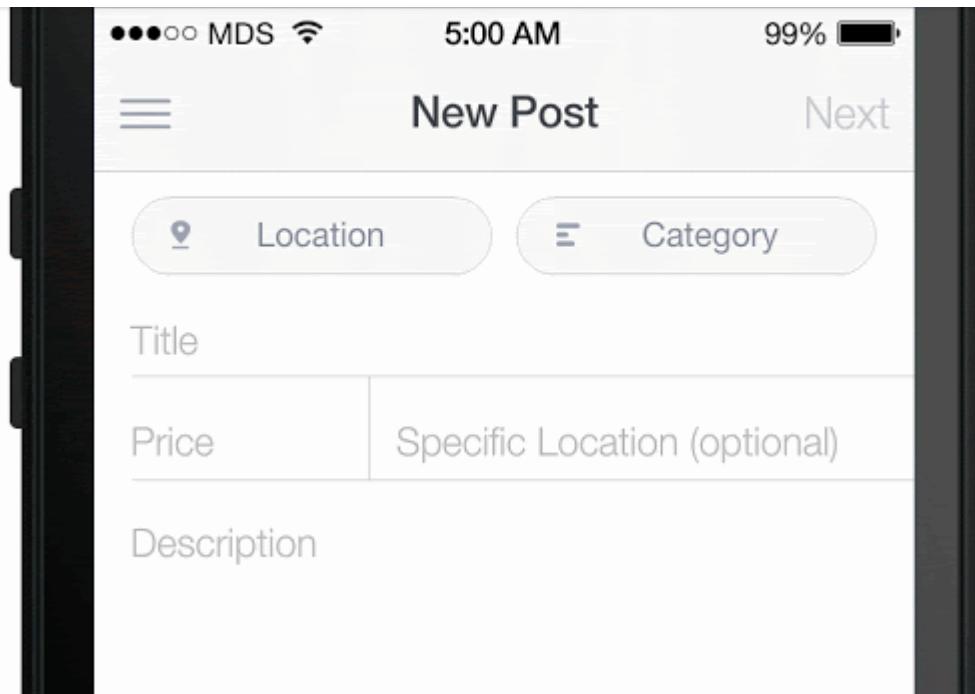
browsers (e.g. Chrome) with auto-translation features and some screen readers don't announce them.



Placeholder text as field label is difficult to read and some screen readers don't announce them.

So why do we use placeholders? They're generally used because visually, they save a lot of space. But problems occur because they tend to disappear once the user clicks the text box. Users then cannot double-check to ensure they've entered their data in the correct field.

One common solution is the floating label, which uses the label as the placeholder. The label is floated above the input field after the user focuses on the form field or begins to type. People tend to prefer this approach as a way to save space, and overall it's a very elegant pattern.



The float label form interaction by [Matt D Smith](#).

However, the 'cons' of floating labels outweigh any 'pros'. First, there is no space for hint text because the label and placeholder are the same. Second, the small nature of the label and its poor contrast (which is a necessity because it allows users to differentiate between a real value and a placeholder) makes it difficult to read. Lastly, they don't really save much space because the label needs space to move into.

What can we do instead? We can position hint text above the input field (see image below). This way, it will still be read by screen readers, and it will also create a larger touch size.

Password

Must contain 8+ characters with at least 1 number and 1 uppercase letter

A form field for a password. Above the input field, the word "Password" is displayed in bold. Below it, a descriptive message states "Must contain 8+ characters with at least 1 number and 1 uppercase letter". The input field itself is a large, empty rectangular box with a black border.

Hint text that is above the input field from Adam Silver's book: [Form Design Patterns](#).

5. Marking optional or required fields*

By convention, we're used to seeing required fields being marked with a red asterisk. But if you're using the aforementioned question protocol, then most of the fields that you're asking for on your form should be required; therefore, it would make more sense to only mark the ones that are optional. When our intention is for something to stand out, we want to provide a visual cue that indicates that it's different.

The image displays two side-by-side wireframe forms. The left form has three input fields, each preceded by a label with a red asterisk. Below these fields is a red rectangular button containing the text "Really, don't do that". The right form also has three input fields, but their labels are different: the first two are just "LABEL", while the third is "LABEL - Optional". Below these fields is a green rectangular button containing the text "Do".

Using the term 'Optional' after a label provides more clarity for users who may not understand what an asterisk means. [Source](#)

While you and I may easily understand that a red asterisk indicates a required field, there is always someone who will wonder what it means. Adding the word 'optional' next to a label is much clearer than relying on a visual symbol.

A quick accessibility note about using the asterisk as a required-field marker: It's possible to [add markup to the form field so that screen readers say the word "required."](#)

6. Display error feedback

When it comes to informing users about an error associated with their submission, there are three aspects of the interface that need to be updated.

Within the HTML document there is an element called <title> which defines the title of the document and it is shown in the browser's title bar or in the page's tab (i.e. in the image above). The <title> tag is the first part of a web page that is read out by screen readers which is why it is such an efficient way to quickly let users know that there is something wrong with their submission.

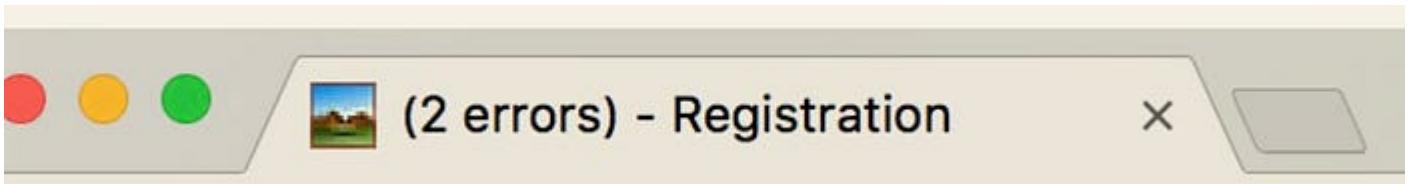


Photo of a browser tab displaying 2 errors from Adam Silver's book: [Form Design Patterns](#).

This solution will require a bit of Javascript (to count the number of errors on the form) so it is not likely that you will be responsible for this as a designer – just make sure you communicate this to the developer.

Validation Summary

This type of feedback is much more visible. A summary of errors at the top of the form lets users know what specifically needs to be fixed.

Form heading

❗ Form contains following errors

- Error 1: [Please enter your first name](#)
- Error 2: [Please enter your last name](#)

First segment heading

First name

❗ Please enter your
first name

Last name

❗ Please enter your last
name

Submit

Error messages summarized at the top of the form.

What's the difference between an average and a good validation summary? A good one contains error links rather than just error messages. Error links allow users to quickly jump to the part of the form that was filled out incorrectly, rather than requiring them to read the message at the top, then have to scroll down until they find where their error was, requiring them to memorize the error message.

Of course, where there are no errors on the form, the summary panel can be hidden.

Inline Errors

In an ideal world, all validation would be inline. This means that as soon as a user has finished filling (or not filling) out a field, there would be an indication nearby if that field contained an error. This can reduce the need for the user to scroll up and down the page to check for any error message.

Last name

 **Last name is required**

Inline error message with red error text and accompanied with an icon just below the field.

There has also been growing popularity with placing the inline error message above the field. This is because the auto-complete panel can obscure the error message when it is below the field. On-screen keyboards can obscure them as well (for mobile).

Password

Must be at least 8 characters

 **Enter a password**

Inline error messages above the field are becoming more common these days.

For inputs that are a little more complex — such as entering a new password — a good solution might be to use instant inline validation, which provides the user with instant feedback while they are typing. This is the most efficient way to inform users they've satisfied the form requirements.

Password Reset

Enter your new password for your Slack account.

New Password

A password input field containing five black dots, indicating the user has entered a password. A small lock icon is in the top right corner of the input field.

Very weak

Confirm New Password

A password input field with a lock icon in the top right corner.

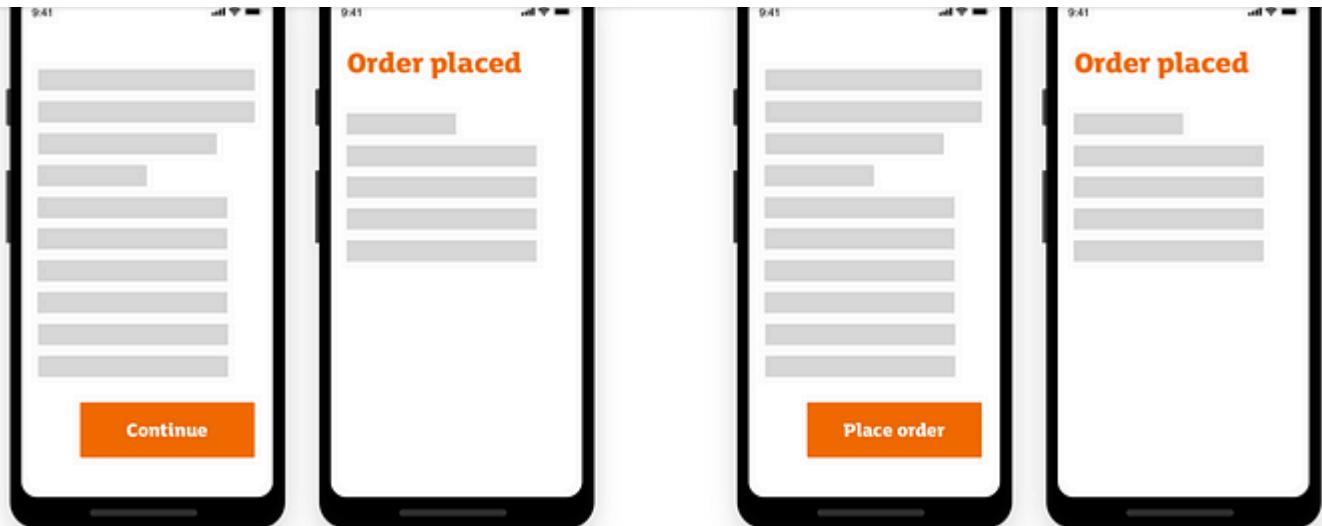
Change my password

Immediate feedback when using instant inline validation.

7. Action buttons

Use action verbs

CTAs should use descriptive rather than generic language. Action verbs help the user understand the action they're going to take when they press a button. While the goal should be to be concise, it shouldn't come at the cost of clarity. Use language that's easy for everyone to understand.



Use action verbs rather than generic language.

Primary vs Secondary

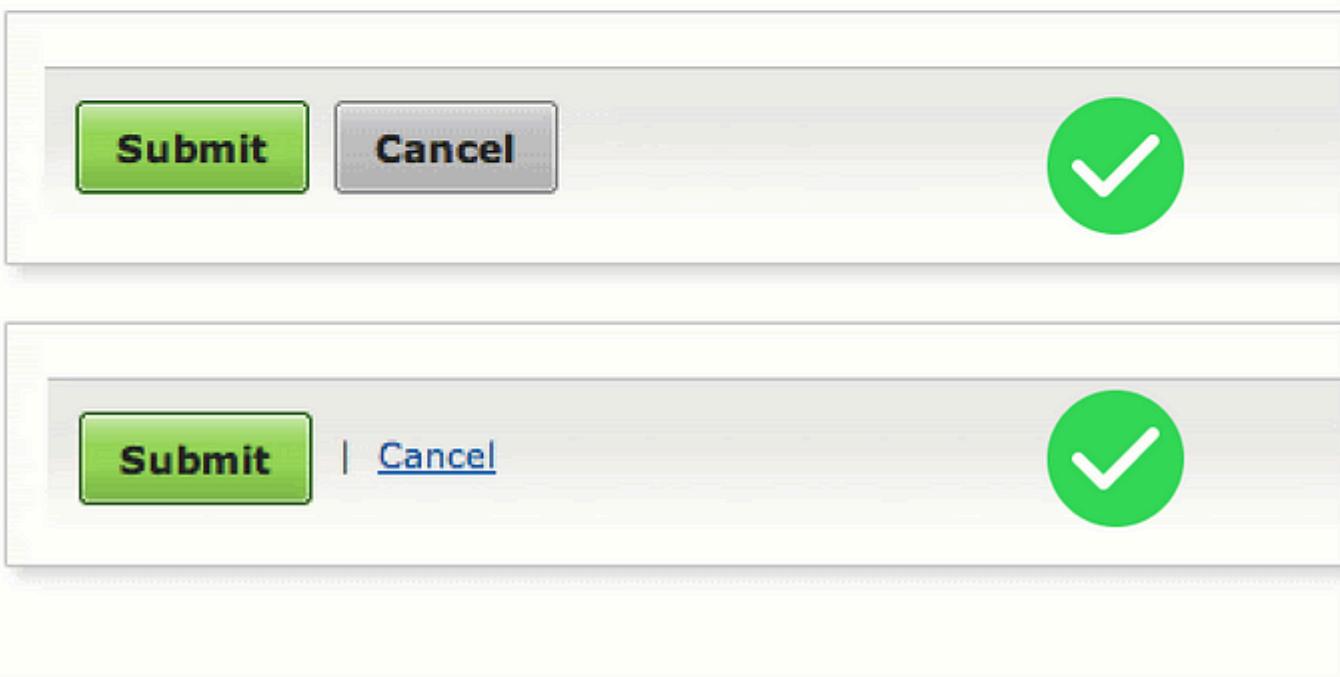
When there is a clear visual distinction between primary and secondary buttons, we mitigate the chances of the user making an incorrect decision. Visual prominence directs users towards the most expected action.

EQUAL VISUAL WEIGHT



This example illustrates the principle of equal visual weight. The 'Submit' and 'Cancel' buttons are visually equivalent in size and color, which can lead to accidental clicks on either one if they are positioned too close together.

VISUAL DISTINCTIONS



Visual distinction reduces the likelihood of an accidental click. In the first example, the 'Submit' button is green and the 'Cancel' button is grey, creating a clear visual distinction. In the second example, the 'Submit' button is green and the 'Cancel' link is blue, also creating a clear visual distinction.

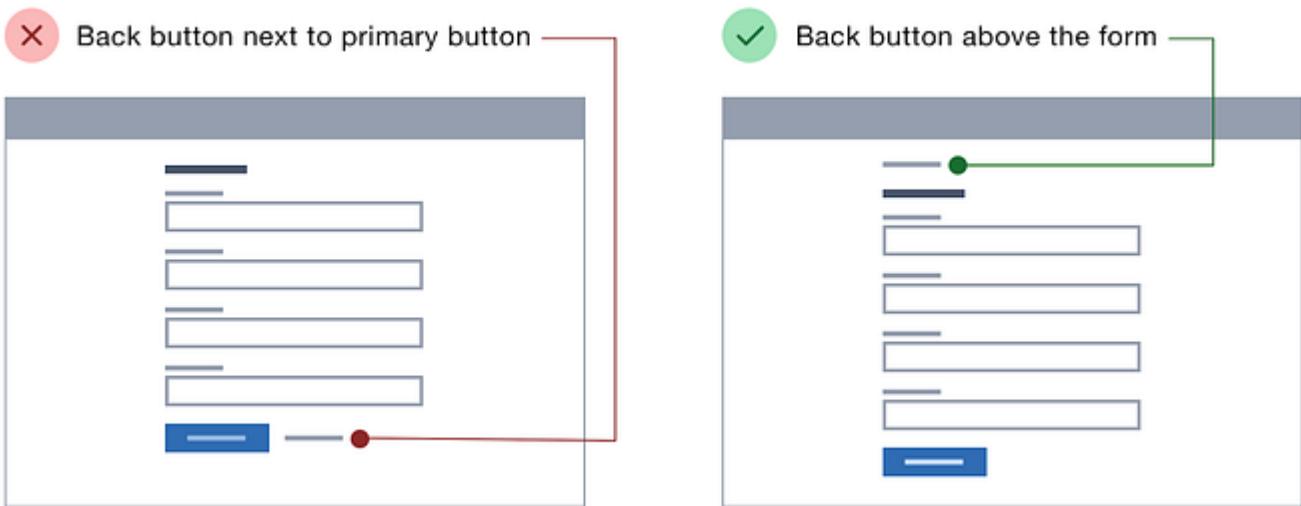
Visual distinction reduces the likelihood of an accidental click. [Source](#)

Placement

Naturally, it makes the most sense for submit buttons to be placed at the bottom of the form because (at least with most forms) users will fill it out from top to bottom and then submit.

Another decision is whether to align the button to the left, right, or center. It makes sense for the submit button to be left-aligned since the labels are also left-aligned and when users look for the next field to complete, they will naturally look down in that location (in left-to-right reading languages).

In forms where a back button is present (i.e. multi-step forms) the submit button can still be left-aligned, but the back button can be placed at the top of the form.



Place the back button (if necessary) at the top of the form. [Source](#)

8. Use multi-step forms when applicable

In situations where there are multiple form fields that can be split up into different categories, using a multi-step form can help improve usability. Organizing fields into several shorter forms will make it more manageable for the user.

The multi-step form works best whenever you need to ask users for a lot of information; Showing their progress is a good way to keep them engaged. Progress indicators display progress sequentially by breaking things up into multiple logical, numbered steps. This gives users clear feedback on how much of the form they've completed and how much more remains. However, avoid numbering the steps unless you are absolutely sure exactly how many steps will be required – and that there will be no deviations.

Cart Shipping Payment Review

Address

11501 Burnet Rd.

Address 2 (optional)

Apartment, unit, suite

City

Austin

State

Texas



Zip code

78751

[Back to cart](#)[Proceed to payment](#)

Example of a multi-step form with a progress indicator near the top.

Test early — and often

Ultimately, it's important to justify the existence of each and every form field. Forms can be tedious tasks for people, and your goal should be to make them as usable as possible. Like other designs, forms can also benefit from testing, so test early and test often, and make any improvements based on the results of your testing.



The UX Collective donates US\$1 for each article we publish. This story contributed to [World-Class Designer School](#): a college-level, tuition-free design school focused on preparing young and talented African designers for the local and international digital product market. Build the design community you believe in.

#inclusive-design

#visual-design

#design

#ux

#technology