Summary of hidden input fields

What are hidden form fields?

Hidden form fields are exactly what they sound like, invisible. This means that users can’t see the field and data when they complete a form.

But don’t let this fool you.

Just because they can’t see them doesn’t mean they aren’t important.

What can you use hidden input fields for?

Many things.

It all depends on your requirements.

Here are a few examples.

#1. Online forms are often used to generate leads. Collecting additional detail about a user can give you more insight into prospects and help you convert more leads into customers in the long run - and this can be done with .... drumroll ... hidden input fields. Whether you’re using Google, Bing or Facebook Ads, or any other PPC platform to generate traffic, these details are often found in hidden form fields.
#2. Hidden input fields may be handy if, for example, you have several forms on different pages on your website and you want to identify which page the visitor was on when they filled out the form.
#3. You may want to include a timestamp of when the user submitted the form.
#4. Another use of hidden inputs is to know what entry a user is editing so that you can update the correct row in the database when the user submits the form. The user doesn't need to edit (or even know) the ID of the entry, so a hidden field works well here.

Alternatives to using the hidden input type

There are many ways to achieve a result when it comes to coding.

So, it shouldn't come as a surprise that you don’t have to use hidden input fields.

#1 URL params

An alternative is to put added information in URL parameters. This can be done by building the parameters into the URL that the form is being submitted to, like this for example:

```
<form action="process.php?user_id=1234">
```

This has its drawbacks.

You have to build the URL properly and escape the data yourself, and the length of URLs servers will accept is limited so it may not work for longer data.

#2. Session variables

If the user wants to edit a page, you could store the user_ID in a session variable, and then retrieve it on the page that saves the changes.

But this also has its downsides.

Setting up and maintaining sessions may require adding code in different places. The user's session could expire in between loading and saving, and you have to make sure it works if the user has multiple windows or tabs open. You also need to make sure it doesn't do weird things when a user hits back or forward. Because of all these potential pitfalls, it is often better to just pass non-sensitive data via a hidden input field.

#3. Cookies

Finally, you can use cookies.

In many languages/frameworks sessions are tracked using cookies, so they're basically the same solution as we spoke about above.

Are hidden inputs safe?

None

Nope.

Remember, anyone can inspect an input element and find out the information that you've decided to store in that field.

If you don’t mind people finding out an ID for example, then it’s not a big deal. But if you are keeping a secret value which you don’t want to share with anyone then you must not use hidden input fields. To overcome this safety issue, you could first encrypt or hash the data and then store them in a hidden input. But a better way is to probably use session variables which we spoke about earlier.

Final comments

- Hidden inputs are invisible to the user

- You create a hidden input like this: `<input type="hidden">`
- A hidden input field gives you the ability to include data that cannot be seen or modified by users when a form is submitted.

- There are tons of reasons why you may want to use a hidden input field (analytics, database references, page tracking, submitting security tokens, passing information from one page to the next, keeping a timestamp, etc.)

- Hidden inputs do not guarantee that the data is secure, so be careful with the type of information you want to store in here.