

# Cognito Forms Integration

HandL UTM Grabber / Tracker How to collect and track UTM variables via Cognito Forms step by step

- [Cognito Form: Embed script implementation](#)
- [Cognito Form: Capture UTMs using Iframe](#)

# Cognito Form: Embed script implementation

First, make sure all the UTM fields are created in Cognito Form. Each field should be separate and Field Name should match the name you will use later in the implementation (see the image below).

The image shows the Cognito Form configuration interface. On the left, the 'Field Settings - Textbox' panel is visible, showing the following settings:

- Label:** utm campaign
- Field Name:** Utmcampaign
- Type:** Single Line (selected)
- Placeholder Text:** test
- Format Validation:** (empty dropdown)
- Default Value:** (empty field)

On the right, the form preview is shown. It includes a title 'test', a label 'testing', and an 'Email' field. Below the email field is a 'utm campaign' field with the placeholder text 'A test'. A context menu is open over the 'utm campaign' field, showing options: Cut, Copy, Delete, Insert Field, Make Smaller, Make Bigger, and Justify Row. Below the form fields is a 'Submit' button and a message: 'Thank you for filling out the form. Your response has been recorded.'

Your default embed code from Cognito Form should look like this

```
<script src="https://services.cognitoforms.com/s/<YOUR FORM ID>"></script>
<script>Cognito.load("forms", { id: "2"});</script>
```

Modify the embed code from Cognito Forms like the following. Notice that, we are adding entry fields to the `Cognito.load`.

```
<script src="https://services.cognitoforms.com/s/<YOUR FORM ID>"></script>
<script>Cognito.load("forms", { id: "2",
entry: {
  ["UTMCampaign":Cookies.get("utm_campaign"),
```

```
"UTMSource":Cookies.get("utm_source") ,  
"UTMMedium":Cookies.get("utm_medium"),  
"UTMContent":Cookies.get("utm_content"),  
"UTMTerm":Cookies.get("utm_term"),  
"IP": Cookies.get("handl_ip"),    "Organic": Cookies.get("organic_source_str")  
}});</script>
```

Here we only illustrated `utm_*` and `handl_ip`. However you can use all the other parameters very similar way. See the list of all the parameters available [here](#)

### Native WP Shortcodes

**NOTE:** Cognito does not support hidden field in the form out of the box. However you can make a regular text field hidden by creating a condition that never satisfies like this.

The screenshot displays a form configuration interface with a modal window titled "Visible When...". The modal is used to set conditions for when a field is visible. It features a list of conditions: "Email" (selected), "contains", and "123131312asdsad". The conditions are connected by "and" and "or" operators. The modal also includes a "Basic Editor" tab, an "Advanced Editor" tab, and "Cancel" and "Save" buttons. The background shows a form configuration panel with various settings like "Placeholder Text", "Format Validation", "Default Value", "Number of Characters", "Help Text", "Show This Field", "Require This Field", "Read-Only", "Limit Quantity", and "Show Custom Error".

# Cognito Form: Capture UTMs using Iframe

First, make sure all the UTM fields are created in Cognito Form. Each field should be separate and Field Name should match the name you will use later in the implementation (see the image below).

The image shows the Cognito Form interface. On the left is the 'Field Settings - Textbox' panel. It includes fields for 'Label' (utm campaign), 'Field Name' (Utmcampaign), 'Type' (Single Line selected), 'Placeholder Text' (test), and 'Format Validation'. On the right is a preview of the form. It has a title 'test', a label 'testing', an 'Email' field, and a 'utm campaign' field. A context menu is open over the 'utm campaign' field, showing options like Cut, Copy, Delete, Insert Field, Make Smaller, Make Bigger, and Justify Row. Below the form fields is a 'Submit' button and a message: 'Thank you for filling out the form. Your response has been recorded.'

Your default iframe code from Cognito Form should look like this

```
<iframe src="https://www.cognitoforms.com/f/<YOUR FORM ID>?id=2"
style="position:relative;width:1px;min-width:100%;*width:100%;" frameborder="0"
scrolling="yes" seamless="seamless" height="323" width="100%"></iframe><script
src="https://www.cognitoforms.com/scripts/embed.js"></script>
```

Modify the iframe code from Cognito Forms like the following. Notice that, we are adding entry fields to the `Cognito.prefill`.

```
<iframe src="https://www.cognitoforms.com/f/<YOUR FORM ID>?id=2"
style="position:relative;width:1px;min-width:100%;*width:100%;" frameborder="0"
```

```
scrolling="yes" seamless="seamless" height="323" width="100%"></iframe><script
src="https://www.cognitoforms.com/scripts/embed.js"></script>
<script>
Cognito.prefill({
  "Utmcampaign": "[utm_campaign]",
    "UTMSource": "[utm_source]" ,
    "UTMMedium": "[utm_medium]",
    "UTMContent": "[utm_content]",
    "UTMTerm": "[utm_term]",
    "IP": "[handl_ip]",
    "Organic": "[organic_source_str]"
});
</script>
```

Here we only illustrated `utm_*` and `handl_ip`. However you can use all the other parameters very similar way. See the list of all the parameters available here [Native WP Shortcodes](#)

**NOTE:** Cognito does not support hidden field in the form out of the box. However you can make a regular text field hidden by creating a condition that never satisfies like this.

The image shows the Cognito Forms editor interface. A modal window titled "Visible When..." is open, allowing the user to set visibility conditions for a form field. The condition is set to "Email" contains "123131312asdsad". The modal includes a "Basic Editor" tab, an "Advanced Editor" tab, and "Cancel" and "Save" buttons. In the background, the form field configuration panel is visible, showing options for Placeholder Text, Format Validation, Default Value, Number of Characters, Help Text, Show This Field (set to "When" with the condition "Email is filled out"), Require This Field (set to "Never"), Read-Only (set to "Never"), Limit Quantity (set to "Never"), and Show Custom Error (set to "Never").