Add settings to your Discourse theme

- :heavy_plus_sign: Adding settings to your theme
- :symbols: Supported types
  - objects type
- :capital_abcd: Setting description and localizations
  - Multiple languages support
- Additional setting attributes and options
  - Min and max attributes
  - Access to settings in your JS/CSS/Handlebars
  - Setting CSS variables
  - Resolving group membership
- :link: Related Topics

Discourse has the ability for themes to have "settings" that can be added by theme developers to allow site owners to customize themes through UI without having to change any line of code and worry about losing their changes with future updates for the theme.

Themes can also alter certain themeable site settings, for more information on that, see the Themeable site settings topic.

:heavy_plus_sign: Adding settings to your theme

Adding settings to your theme is a bit different from adding CSS and JS code, that is there is no way to do it via the UI.

The way to add settings is to create a repository for your theme, and in the root folder of your repository create a new `settings.yaml` (or `settings.yml`) file. In this file you'll use the YAML language to define your theme settings.

> :loudspeaker: **Note:** You may find it helpful to make use of the Theme CLI, which tremendously simplifies the development process.

Now if you're familiar with plugin development, this shouldn't be a new thing to you - it mostly works the same way as adding site settings to your plugin. Just dump some valid YAML in your settings file and you'll be good to go.

A valid theme setting must have a name and default value, that's the bare minimum and it looks like this:

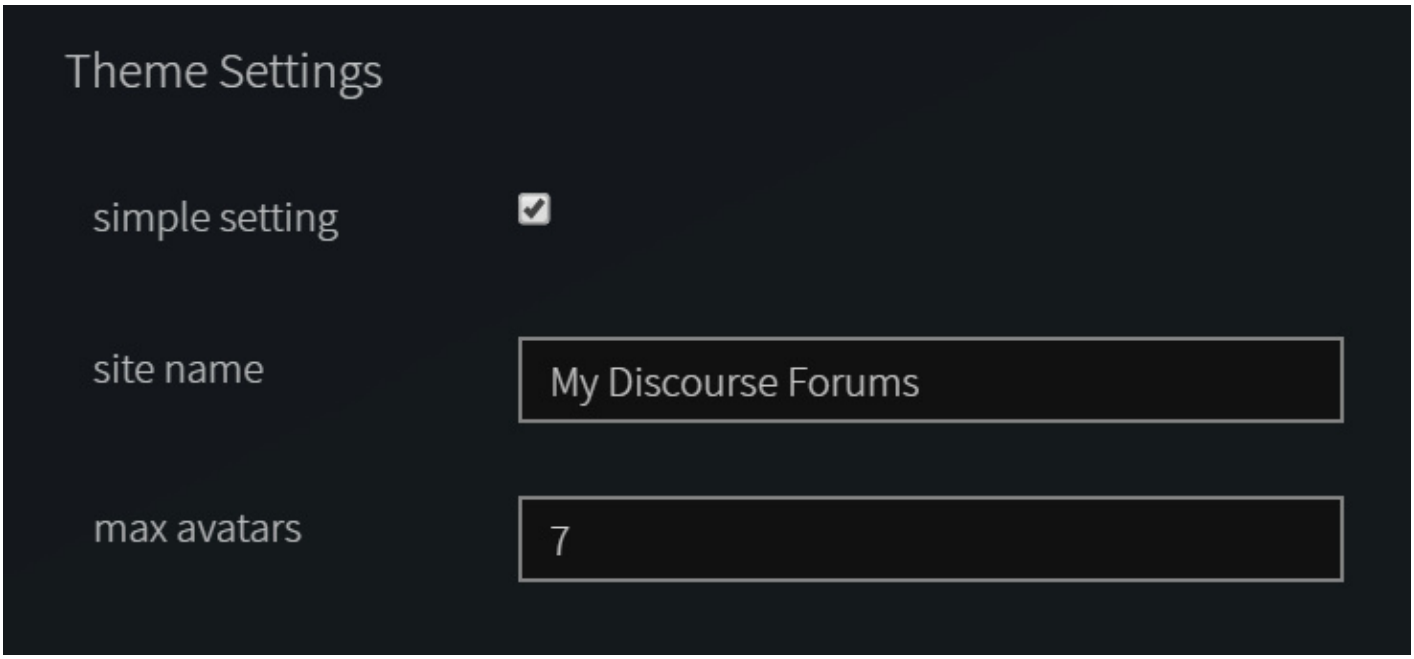
```
simple_setting: true
```

As you can probably tell, that will create a setting with the name `simple_setting` and it'll have `true` as its default value.

Similarly, you can add something like this:

```
site_name: My Forums
max_avatars: 7
```

And you'll have two more settings, `site_name` which will be a string setting with "My Forums" as the default value, and `max_avatars` as an integer setting with default value of 7.



You can access your settings in your JS code like this: `settings.your_setting_key`.

So until this point we've covered the simplest way to define settings. In the next section we'll dive a bit deeper into the various types of settings and how you can use them.

:symbols: Supported types

There are 9 types of settings:

1. `integer`
2. `float`
3. `string`
4. `bool` (for boolean)
5. `list`
6. `enum`
7. `objects` (replacement for `json_schema`)
8. `upload` (for images)
9. `icon` (for a single icon from the Discourse icon set)

And you can specify type by adding a `type` attribute to your setting like this:

```
float_setting:
  type: float
  default: 3.14
```

I should say that you don't always have to explicitly set a `type` attribute because Discourse is smart enough to work out the setting type from the setting's default value. So you can reduce the above example to this:

```
float_setting:
  default: 3.14
```

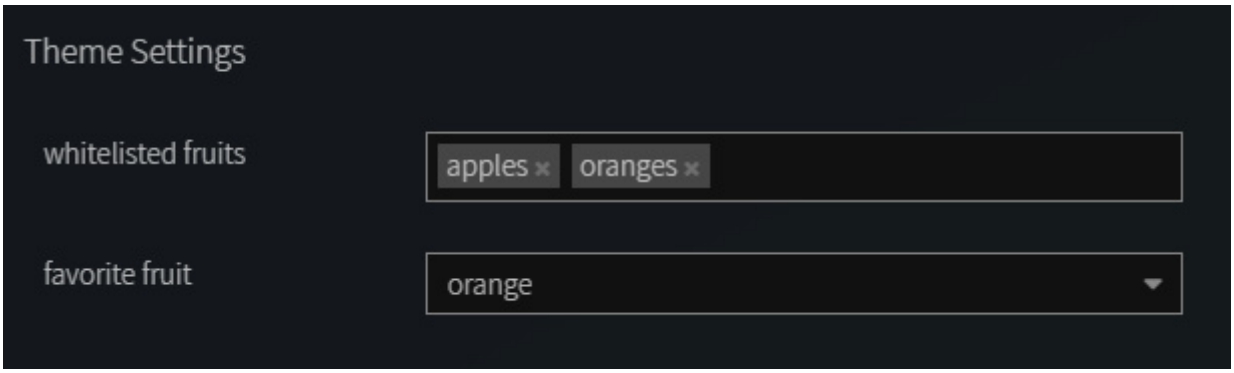
That said, you *need* to set a type attribute when working with **`list`**, **`enum`** and **`icon`** settings, otherwise Discourse will not recognize them correctly.

List Setting:

```
whitelisted_fruits:
  default: apples|oranges
  type: list
```

Enum Setting:

```
favorite_fruit:
  default: orange
  type: enum
  choices:
    - apple
    - banana
```



In case the difference between list and enum settings is not clear to you: enum settings allow your theme users to select only *one* value from a set of values defined by you (see the `choices` attribute).

On the other hand, list settings allow your users to create their own *list* (i.e. an array) of values. They can add to or remove from the setting's default list of values. You can set the default list of values for the setting by joining the values with a vertical bar | character. See the list setting in the example above.

You can see a real-world use case for list settings here: https://meta.discourse.org/t/linkify-words-in-post-theme-component/82193?u=osama.

> :loudspeaker: **Note**: Pay attention to indentation when working with YAML because YAML is very picky about spaces and will throw a syntax error if your code indentation is incorrect.

Icon Setting:

```
banner_icon:
  default: bullhorn
  type: icon
```

Icon settings give site owners a searchable icon picker, and the value is the icon name. Discourse adds the selected icon to the sprite sheet, so you can render it in your theme without registering it separately.

`objects` type

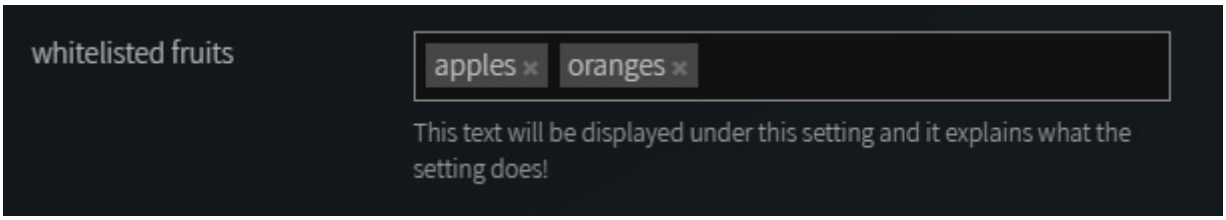
The `objects` setting type is a special type that allows you to accomplish advanced settings with custom structure and validations. We have a separate documentation for this type.

:capital_abcd: Setting description and localizations

You can add description text to your theme setting and it'll be shown as a label directly under the setting. To do that simply add a `description` attribute to your setting like so:

```
whitelisted_fruits:
  default: apples|oranges
  type: list
  description: "This text will be displayed under this setting and it explains what the setting does!"
```

And you'll get this:



Multiple languages support

If you know more than one language, and you'd like to add support for those languages to your theme, then you can totally do that provided that Discourse supports said languages.

First of all, make sure the language you want to support is in this list:

[details="Languages list"]

Code				Name
ar				اللغة العربية
bs_BA				bosanski jezik

Code				Name
ca				català
cs				čeština
da				dansk
de				Deutsch
el				ελληνικά
en				English
es				Español
et				eesti
fa_IR				فارسی
fi				suomi
fr				Français
gl				galego
he				תירבע
id				Indonesian
it				Italiano
ja				□□□
ko				□□□
lv				latviešu valoda
nb_NO				Norsk bokmål
nl				Nederlands
pl_PL				język polski
pt				Português
pt_BR				Português (BR)
ro				limba română
ru				Русский
sk				slovenčina
sq				Shqip
sr				српски језик
sv				svenska
te				□□□□□
th				□□□

Code				Name
tr_TR				Türkçe
uk				українська мова
ur				ودرا
vi				Việt Nam
zh_CN				□□
zh_TW				□□ (TW)

[/details]

(If you can't see your language in the list then you might want to take a look at https://meta.discourse.org/t/how-to-add-a-new-language/14970?u=osama)

Then you'll need find your language code from the above list and use the language code as a key under the `description` attribute and translation as a value for the key like so:

```
whitelisted_fruits:
  default: apples|oranges
  type: list
  description:
    en: English text
    ar: نص باللغة العربية
    fr: Texte français
```

And now you have support for 3 languages: English, Arabic and French.

Additional setting attributes and options

Min and max attributes

Sometimes you may need to specify limits that a setting value can't exceed to prevent your users from accidentally breaking the theme or possibly the whole site.

To specify limits, simply add a `min` or `max` or both attributes to your setting like so:

```
integer_setting:
  default: 10
  min: 5
  max: 100
```

You can specify limits to `integer`, `float` and `string` type settings. For `integer` and `float` settings, the value of the setting itself is checked against the limits. And for `string` settings, the length of the value is checked against the specified limits.

If your user tries to enter a value that's not within the allowed range, they'll see an error telling them what the min and max values are.

Access to settings in your JS/CSS/Handlebars

Theme settings are made available globally as a `settings` variable in theme JavaScript files. For example:

```
// {theme}/javascripts/discourse/api-initializers/init-theme.gjs
import { apiInitializer } from "discourse/lib/api";

export default apiInitializer((api) => {
  console.log("settings are", settings);
});
```

This `settings` object is also usable as normal within `.gjs` `<template>` tags.

Setting CSS variables

In CSS, you'll get a variable created for every setting of your theme and each variable will have the same name as the setting it represents.

So if you had a float setting called `global_font_size` and a string setting called `site_background`, you could do something like this in your theme CSS:

```
html {
  font-size: #{$global-font-size}px;
  background: $site-background;
}
```

Resolving group membership

Theme components sometimes need to show or hide a feature based on whether the current user is in a configured group. Avoid checking `currentUser.groups` for this because it only includes groups that are visible to the user, and it can miss hidden groups.

For group-backed list settings, add `resolve_group_membership: true` to resolve the check server-side:

```
copy_button_allowed_groups:
  default: "1|3"
  type: list
  list_type: group
  resolve_group_membership: true
```

This option is only valid when the setting has `type: list` and `list_type: group`. When it is enabled, the frontend `settings` object does not include the original group list. Instead, Discourse adds a boolean with the same setting name prefixed by `user_in_`:

```
// {theme}/javascripts/discourse/api-initializers/init-theme.gjs
import { apiInitializer } from "discourse/lib/api";

export default apiInitializer((api) => {
  if (!settings.user_in_copy_button_allowed_groups) {
    return;
  }

  // User is in at least one selected group.
});
```

The generated boolean also works with automatic groups such as `logged_in_users` and `anonymous_users`. Object theme settings can use the same option on `type: groups` properties. See objects type for theme settings for details.

:link: Related Topics

- https://meta.discourse.org/t/developer-s-guide-to-discourse-themes/93648

□□□ #3
□ □□□ □□□ 2026-10-31 13:30:41 UTC
□ cocobook-sync □□□ 2026-11-06 07:04:04 UTC