Users

- Get user by field
- Twig
- Invalid user
- Login state
- User data
- Extending Timber\User
- Querying Users

To get a user object in Timber, you use `Timber::get_user()` and pass the WordPress user ID as an argument.

```
$user = Timber::get_user($user_id);
```

This function is similar to `get_userdata()` and accepts one argument: a user ID. If you don't pass in any argument, Timber will use `get_current_user_id()` to work with the currently logged in user.

```
$user = Timber::get_user();

// Is the same as…

$user = Timber::get_user(get_current_user_id());
```

What you get in return is a `Timber\User` object, which is similar to `WP_User`.

Get user by field

If you don't have a user ID, you can also get a user by other fields, like `email` or `login` through `Timber::get_user_by()`.

```
// Get a user by email.
$user = Timber::get_user_by('email', 'user@example.com');

// Get a user by login.
$user = Timber::get_user_by('login', 'keanu-reeves');
```

Twig

You can convert user IDs to user objects in Twig using the `get_user()` function.

```
{% set user = get_user(user_id) %}
```

It also works if you have an array of user IDs that you want to convert to `Timber\User` objects. Use the `get_users()` function.

```
{% for user in get_users(user_ids) %}

{% endfor %}
```

Invalid user

If no user can be found with the user ID you provided, the `Timber::get_user()` function will return `null`. With this, you can always check for valid users in a template a simple if statement.

```
$user = Timber::get_user($user_id);

if ($user) {
    // Handle user.
}
```

Or in Twig:

```
{% if user %}
    {{ user.name }}
{% endif %}
```

Login state

Similar to checking for valid users, you can also check whether a user is currently logged in to WordPress. When you don't provide an ID for `Timber::get_user()`, it will return `null` if no user is currently logged in.

```
$user = Timber::get_user();

if ($user) {
    // A user is logged in.
} else {
    // No user is logged in.
}
```

This allows you to check for a login state with an if statement.

```
{% if user %}
    Hello {{ user.name }}!
{% else %}
    Hello visitor!
{% endif %}
```

User data

If you need to access user data like the first and the last name, you can either load them from the properties set on user or with `{{ user.meta }}` for user values that are saved as meta values.

Here's a little cheat sheet:

```
{# Display name #}
{{ user.name }}
{{ user.display_name }}

{# Nice name #}
{{ user.slug }}
{{ user.user_nicename }}

{# Email #}
```

```
{{ user.user_email }}

{# Website #}
{{ user.user_url }}

{# Values loaded from user meta. #}
{{ user.meta('first_name') }}
{{ user.meta('last_name') }}
{{ user.meta('description') }}

{# Contact info #}
{{ user.meta('facebook') }}
{{ user.meta('instagram') }}
{{ user.meta('linkedin') }}
{{ user.meta('pinterest') }}
{{ user.meta('soundcloud') }}
{{ user.meta('tumblr') }}
{{ user.meta('twitter') }}
{{ user.meta('youtube') }}
{{ user.meta('wikipedia') }}
```

Extending `Timber\User`

If you need additional functionality that the `Timber\User` class doesn't provide or if you want to have cleaner Twig templates, you can extend the `Timber\User` class with your own classes:

```
class Author extends Timber\User
{
}
```

To initiate your new `Author` user, you also use `Timber::get_user()`.

```
$author = Timber::get_user($user_id);
```

You **can't** instantiate a `Timber\User` object or an object that extends this class with a constructor – you can't use `$author = new Author( $user_id )`. In Timber, we've chosen to go a different way to prevent a lot of problems that would come with direct instantiation.

So, how does Timber know about your `User` class? Timber will use the User Class Map to sort out which class it should use.

Querying Users

If you want to get an array of users, you can use `Timber::get_users()`.

```
$users = Timber::get_users($query);
```

You can use this function in a similar way to how you use `WP_User_Query`.

```
// Get all users that only have a subscriber role.
$subscribers = Timber::get_users([
    'role' => 'subscriber',
]);

// Get all users that have published posts.
$post_authors = Timber::get_users([
    'has_published_posts' => ['post'],
]);
```

Instead of a query, you can also pass an **array of user IDs**.

```
$users = Timber::get_users([27, 83, 161]);
```

If you don't pass in any argument, `Timber::get_users()` will do nothing and return an empty array.

```
// Returns an empty array.
$users = Timber::get_users();
```

□□□ #2
□ □□□ □□□ 2026-08-29 16:14:01 UTC
□ □□□ □□□ 2026-08-29 16:14:01 UTC