Template Engines

A template engine is software that combines pre-defined templates with dynamically generated data and is often used by web applications to generate dynamic responses. An everyday use case for template engines is a website with shared headers and footers for all pages. A template can dynamically add content but keep the header and footer the same. This avoids duplicate instances of header and footer in different places, reducing complexity and thus enabling better code maintainability. Popular examples of template engines are Jinja and Twig.

Templating

Template engines typically require two inputs: a template and a set of values to be inserted into the template. The template can typically be provided as a string or a file and contains pre-defined places where the template engine inserts the dynamically generated values. The values are provided as key-value pairs so the template engine can place the provided value at the location in the template marked with the corresponding key. Generating a string from the input template and input values is called `rendering`.

The template syntax depends on the concrete template engine used. For demonstration purposes, we will use the syntax used by the `Jinja` template engine throughout this section. Consider the following template string:

Code: jinja2

```
Hello {{ name }}!
```

It contains a single variable called `name`, which is replaced with a dynamic value during rendering. When the template is rendered, the template engine must be provided with a value for the variable `name`. For instance, if we provide the variable `name="vautia"` to the rendering function, the template engine will generate the following string:

```
Hello vautia!
```

As we can see, the template engine simply replaces the variable in the template with the dynamic value provided to the rendering function.

While the above is a simplistic example, many modern template engines support more complex operations typically provided by programming languages, such as conditions and loops. For instance, consider the following template string:

Code: jinja2

```
{% for name in names %}
Hello {{ name }}!
{% endfor %}
```

The template contains a `for-loop` that loops over all elements in a variable `names`. As such, we need to provide the rendering function with an object in the `names` variable that it can iterate over. For instance, if we pass the function with a list such as `names=["vautia", "21y4d", "Pedant"]`, the template engine will generate the following string:

```
Hello vautia!
Hello 21y4d!
Hello Pedant!
```

Previous Next

Mark Complete & Next

Cheat Sheet

Table of Contents

My Workstation

OFFLINE

Start Instance

∞ / 1 spawns left