**Presentation: Code Generation and Templating (Jinja2)**

Team 3 in INFO 7390

Team Members: Hongwei Hu, Guangnan Liang

GitHub: <https://github.com/Besimilar/Code-Generation-And-Templating>

Resources:

1. Jinja2 Demo Docker Image: besimilar/advanced-data-analysis:template
2. Material Source: <https://www.fullstackpython.com/template-engines.html>

Note: Red lines means Very Important Steps

**Part1: Template Engine**

* Template engines allow developers to generate desired content types, such as HTML, while using some of the data and programming constructs such as if-else and for loops to manipulate the output.
* Template files that are created by developers and then processed by the template engine consist of prewritten markup and template tag blocks where data is inserted.
* The template engine would generate the HTML output response when an HTTP request comes in for a particular URL.

**Part2: Jinja2**

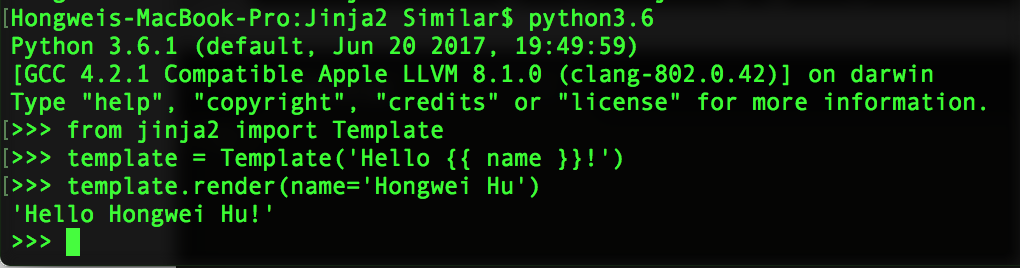
1. Jinja, also known and referred to as "Jinja2", is a popular Python template engine written as a self-contained open source project.
2. Features (<http://jinja.pocoo.org/docs/2.9/>):

* powerful automatic HTML escaping system for cross site scripting prevention.
* Template inheritance makes it possible to use the same or a similar layout for all templates.
* High performance with just in time compilation to Python bytecode. Jinja2 will translate your template sources on first load into Python bytecode for best runtime performance.
* Easy to debug with a debug system that integrates template compile and runtime errors into the standard Python traceback system.
* …

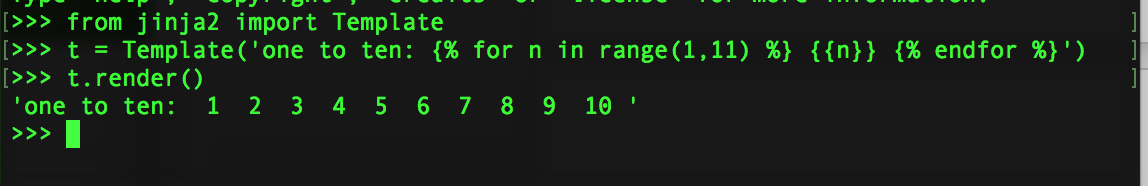
1. Installation

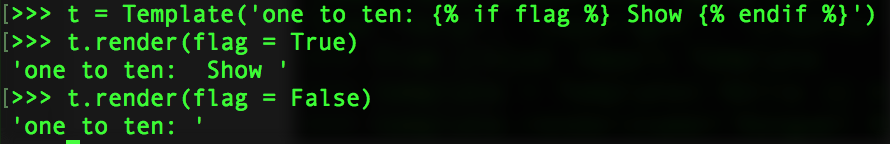
* $ pip install Jinja2

1. Demo
   * Python API ('src/basicAPI-demo/basicAPI-1.py')
2. By creating an instance of **Template** you get back a new template object that provides a method called **render().**
3. Call render() with a **dict** or **keyword arguments** to fill the template. The dict or keywords arguments passed to the template are the so-called “context” of the template.



1. We can also use for loops OR if-else in a template.

****

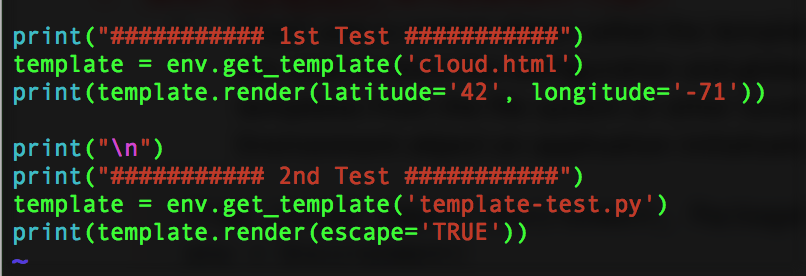
****

* Basic ('src/basicAPI-demo/basicAPI-2.py')
  1. Jinja2 uses a central object called the template Environment. Instances of this class are used to store the configuration and global objects, and are used to load templates from the file system or other locations. Most applications will create one Environment object on application initialization and use that to load templates.



This will create a template environment with the default settings and a loader that looks up the templates in the templates folder inside the yourapplication python package.

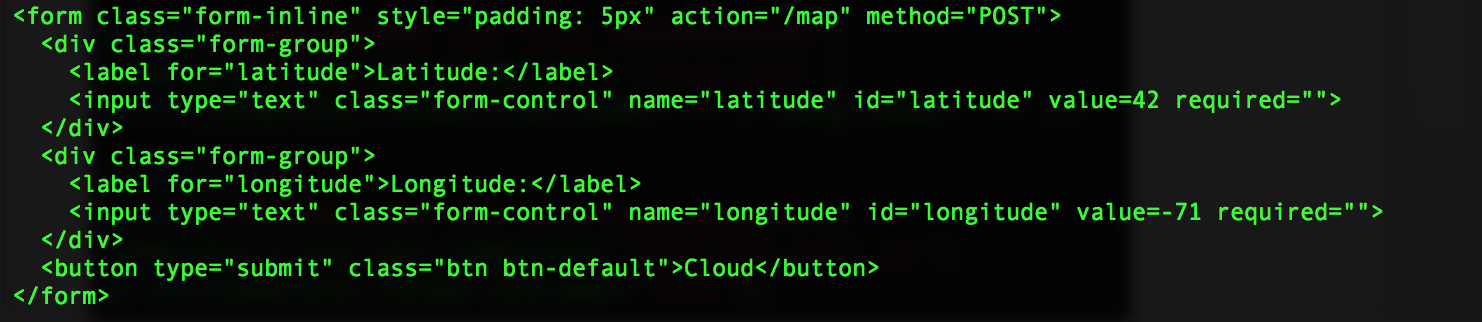
* 1. All parameters inside {{ }} in the templates will be replaced by their values from render().



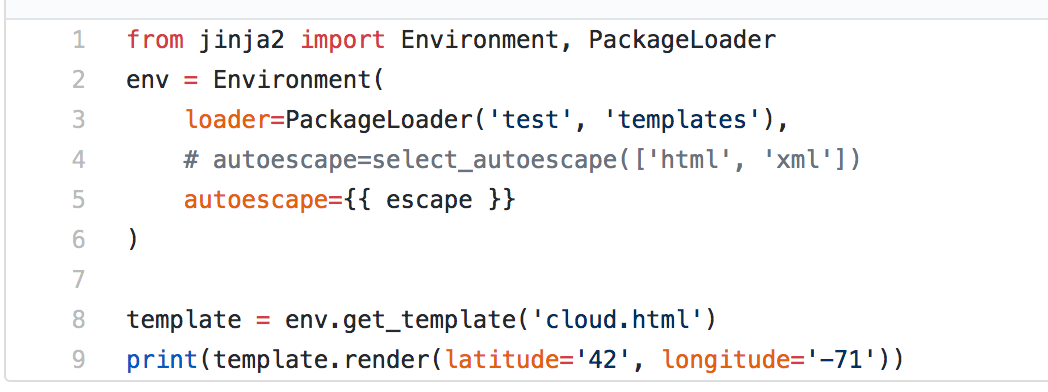
* 1. For “cloud.html”:
     + Before



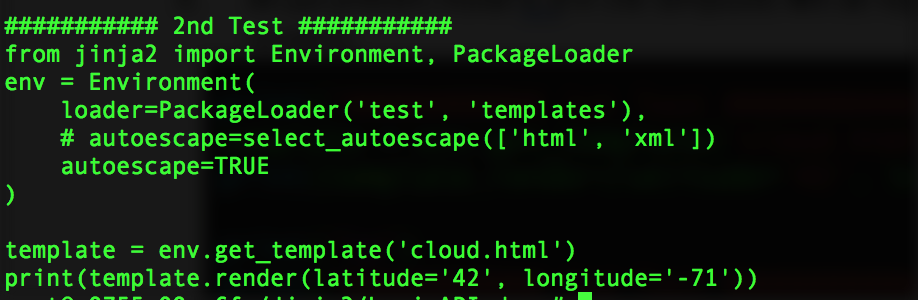
* + - After



* 1. For “template-test.py”
     + Before

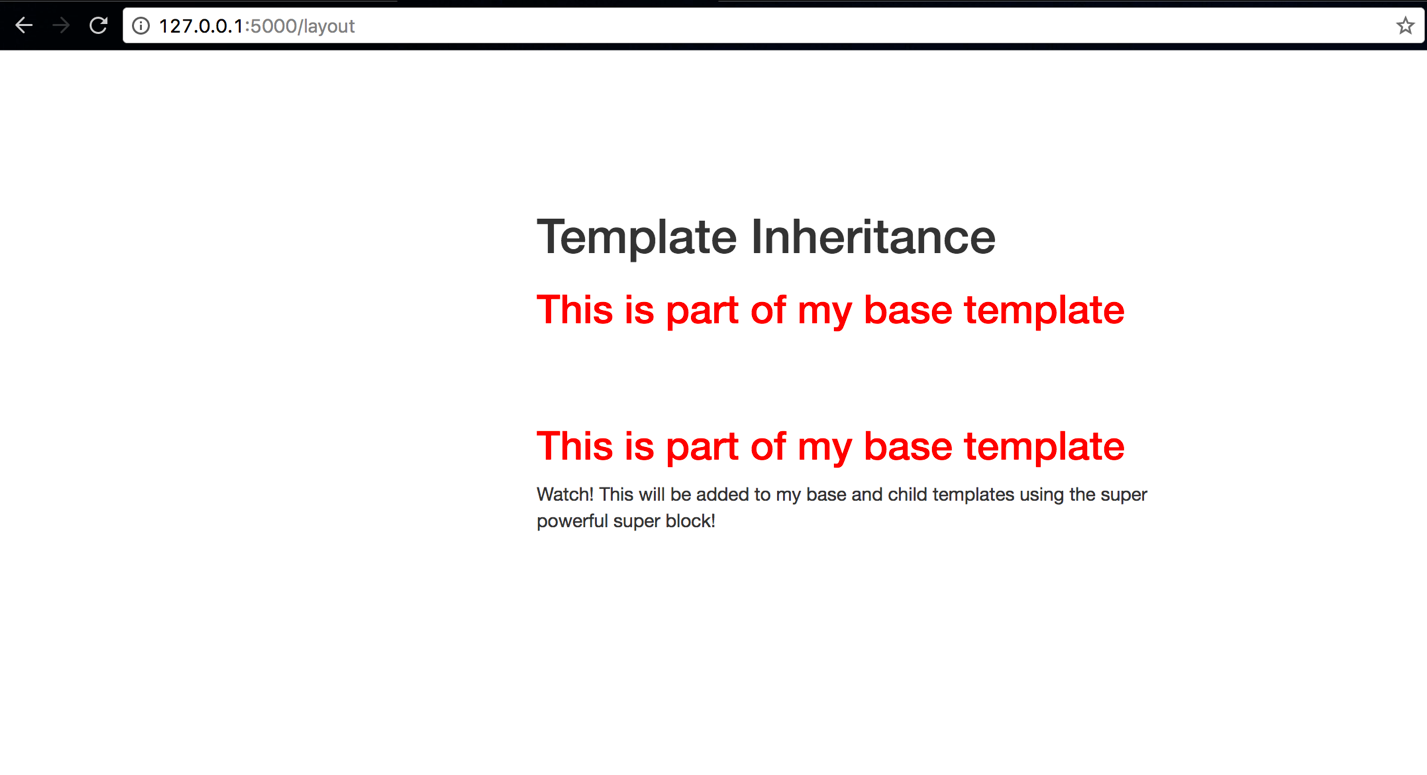


* + - After

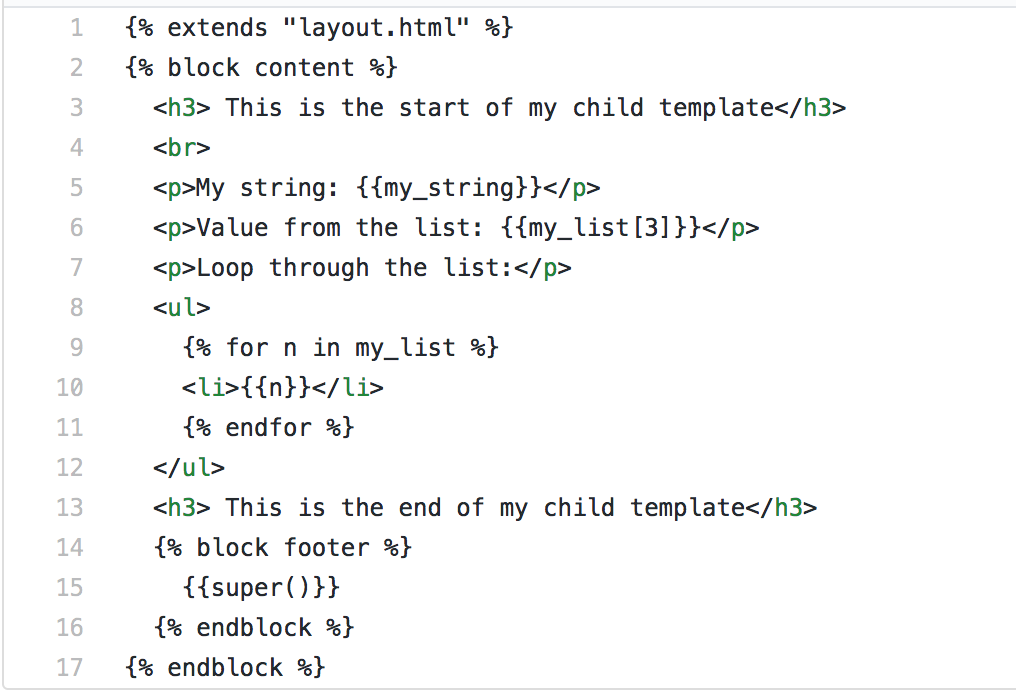


* Flask-demo: ('src/Flask-demo/\*')
  1. Run Flask REST API: $ python3.5 run.py
  2. Create a layout (“layout.html”)





* 1. Create a child template to inherit the layout using {% extends %} (“template.html”)



* 1. How to pass value and render (“run.py”):

