

# Flask Madlibs



[Download our Starter Code](#)

In this exercise, you'll use Flask to make a Madlibs game.

## About Madlibs



In Madlibs, you're asked a series of questions, like this:

*plural\_noun*: **turnips**  
*verb*: **juggle**

Those are then plugged into a story template, like this:

I love to {verb} {plural\_noun}.

To create a story like:

I love to juggle turnips.

## Code

We've given you some code to help with the core non-Flask-specific Madlibs part:

```
stories.py

"""Madlibs Stories."""

class Story:
    """Madlibs story.

    To make a story, pass a list of prompts, and the text
    of the template.

    >>> s = Story(["noun", "verb"],
    ...          "I love to {verb} a good {noun}.")

    To generate text from a story, pass in a dictionary-like thing
    of {prompt: answer, promp:answer):

    >>> ans = {"verb": "eat", "noun": "mango"}
    >>> s.generate(ans)
    'I love to eat a good mango.'
    """

    def __init__(self, words, text):
        """Create story with words and template text."""

        self.prompts = words
        self.template = text

    def generate(self, answers):
        """Substitute answers into text."""

        text = self.template

        for (key, val) in answers.items():
            text = text.replace("{} + key + {}".format(key, val))

        return text

# Here's a story to get you started

story = Story(
    ["place", "noun", "verb", "adjective", "plural_noun"],
    """Once upon a time in a long-ago {place}, there lived a
    large {adjective} {noun}. It loved to {verb} {plural_noun}."""
)
```

This allows you to define Madlibs stories, and it can generate the resulting story from a set of answers. (It's also a nice example of a small but useful class!)

We've created a story, **story**, in that file.

### Warning: STOP AND EXPLORE HERE

Before starting to make a Flask app, make sure you understand how this **Story** class works — go into *ipython* and try out the **generate** method on our sample story to get a feel for the text-generating process for Madlibs.

## Challenge

Write a Flask app that imports the example story. Add a homepage for the application that shows a form prompting you for all the words in the story:

Madlibs

place:

noun:

verb:

adjective:

plural\_noun:

Submit

**Don't hardcode this, though** — you want your form route to be able to ask for all of the questions required by the story, not for it to have a hard-coded form of asking these exact questions!

Add a route, **/story**, that shows the resulting story for those answers, like this:

Your Story

Once upon a time in a long-ago condo, there lived a large frozen dragon. It loved to eat popsicles.

For now, don't worry about having template inheritance or a **base.html** — later, in further study, you can refactor this to use template inheritance.

## Further Study

### Use Template Inheritance

Make a **base.html** template of common parts of your templates (like the **<html>**, **<body>**, and other common things, and change your templates so they inherit from this base template.

### Allow User to Pick Story

Add a feature where there are several different story templates, rather than just one.

The homepage should change to a drop-down menu of the story templates. When the user picks a template, it should go to the page that prompts for the list of story questions. That should, as before, go to the page that shows the generated story.

### Add CSS

**Still want more?** Add some CSS to your madlibs, storing the CSS file in a **static/** directory and referencing it properly, so Flask will serve it up.

### Additional Further Study

**What? More time?** Add some JS to your madlibs — perhaps you can validate the form (make sure every question is answered, all answers are at least 3 characters long, all lowercase, etc) before you're allowed to submit the form.

**Even more, you say???** Try to add a page to your application where uses can create their own madlibs, by providing a list of parts of speech, along with the text of the story. Submitting this form should create a new story instance that you could then select from the dropdown of stories.

**Note: This will be challenging given what you know now!** Consider it a super bonus. Also, because we don't yet know how to persist data, if you store you stories on the server in a list, that list will get reset every time you restart your server.

## Solution

[View Our Solution](#)