**Working with Forms**

**HTML forms**

In HTML, a form is a collection of elements inside <form>...</form> that allow a visitor to do things like enter text, select options, manipulate objects or controls, and so on, and then send that information back to the server.

**The Django** [**Form**](https://docs.djangoproject.com/en/5.1/ref/forms/api/#django.forms.Form) **class**

At the heart of this system of components is Django’s [Form](https://docs.djangoproject.com/en/5.1/ref/forms/api/#django.forms.Form) class. In much the same way that a Django model describes the logical structure of an object, its behavior, and the way its parts are represented to us, a [Form](https://docs.djangoproject.com/en/5.1/ref/forms/api/#django.forms.Form) class describes a form and determines how it works and appears.

* In a similar way that a model class’s fields map to database fields, a form class’s fields map to HTML form <input> elements. (A [ModelForm](https://docs.djangoproject.com/en/5.1/topics/forms/modelforms/#django.forms.ModelForm) maps a model class’s fields to HTML form <input> elements via a [Form](https://docs.djangoproject.com/en/5.1/ref/forms/api/#django.forms.Form); this is what the Django admin is based upon.)
* A form’s fields are themselves classes; they manage form data and perform validation when a form is submitted. A [DateField](https://docs.djangoproject.com/en/5.1/ref/forms/fields/#django.forms.DateField) and a [FileField](https://docs.djangoproject.com/en/5.1/ref/forms/fields/#django.forms.FileField) handle very different kinds of data and have to do different things with it.
* A form field is represented to a user in the browser as an HTML “widget” - a piece of user interface machinery. Each field type has an appropriate default [Widget class](https://docs.djangoproject.com/en/5.1/ref/forms/widgets/), but these can be overridden as required.

## **Building a form**

Suppose you want to create a simple form on your website, to obtain the user’s name. You’d need something like this in your template:

A computer code on a white background

Description automatically generated

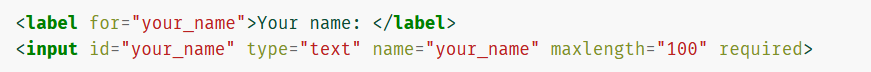
**Building a form in Django**

We already know what we want our HTML form to look like. Our starting point for it in Django is this:

A screen shot of a computer

Description automatically generated

The whole form, when rendered for the first time, will look like:



Note that it **does not** include the <form> tags, or a submit button. We’ll have to provide those ourselves in the template.

**The view**

Form data sent back to a Django website is processed by a **view**, generally the same view which published the form. This allows us to reuse some of the same logic.

To handle the **form**, we need to instantiate it in the **view** for the URL where we want it to be published:

A screenshot of a computer program

Description automatically generated

**The template**

We don’t need to do much in our **name.html** template:

A blue arrow pointing to a blue line

Description automatically generated

**Complete Code:**

**forms.py**

A screen shot of a computer

Description automatically generated

**views.py**

A screenshot of a computer program

Description automatically generated

**name.html**

A blue arrow pointing to a blue line

Description automatically generated

References:

<https://docs.djangoproject.com/en/5.1/topics/forms/>