MAKE SCHOOL

Class-based Forms with Flask-WTForms

BEW 1.2

Agenda

MAKE SCHOOL

- Learning Outcomes
- Warm-Up
- Intro to WTForms
- Lab Activity
- **BREAK**
- Review Concepts

Learning Outcomes

MAKE SCHOOL

By the end of today, you should be able to...

1. **Explain** how using class-based forms can improve your site's security, simplicity, and code readability.
2. **Create** class-based forms with WTForms with a given set of fields.
3. **Use** class-based forms to display form data in a template.

MAKE SCHOOL

Warm-Up Review: Querying the Database

Querying Objects

MAKE SCHOOL

We can query objects with .get() or .filter_by():

```
# Get the user with id=5
my_user = User.query.get(5)
```

```
# Get the user with username "me"
my_user = User.query.filter_by(username="me").one()
```

Querying Many Objects

MAKE SCHOOL

We can get a list of all objects matching a filter with .all() or .filter_by():

```
# Get all users
my_user = User.query.all()
```

```
# Get all users with type "student"
my_user = User.query.filter_by(type="student").all()
```

Creating Objects

MAKE SCHOOL

We can create an object by creating an instance of the model class:

```
# Create a user with username="ilikecoding"
# and password "1234"
my_user = User(username="ilikecoding", password="1234")
```

```
# Save the user to the database
db.session.add(my_user)
db.session.commit()
```

Updating Objects

We can update an object by updating its model instance:

```
# Update the user to have birth date of 1/15/2000
my_user.birth_date = datetime.Date(2000, 1, 15)
```

```
# Save the user to the database
db.session.add(my_user)
db.session.commit()
```

Querying Objects

MAKE SCHOOL

Let's say we have a model **BlogPost** with fields **id**, **title**, **description**, & **author_id**.

Write code to get the **BlogPost** with **id=7** and print its title.

Students, write your response!

Creating Objects

MAKE SCHOOL

Let's say we have a model **BlogPost** with fields **id**, **title**, **description**, & **author_id**. How do we create an instance of it & save to the database?

Write example code to create & save a **BlogPost** with **author_id=1**.

Students, write your response!

Updating Objects

MAKE SCHOOL

Let's say we have a model **BlogPost** with fields **id**, **title**, **description**, & **author_id**.

Write code to update the **BlogPost** object with **id=7** to have the **title** of **"Hello, World"**.

Students, write your response!

MAKE SCHOOL

What is a Form?

What is a Form?

MAKE SCHOOL

With your group, write down everything you know about **forms**.

-
- GET vs. POST
- Jinja2
- Html tags
- Input tags - text, numbers, etc
- Asks user for info.
- Help database with information and responses to user inputs

-
-
-
-
-

What is a Form?

MAKE SCHOOL

With your group, write down everything you know about **forms**.

MAKE SCHOOL

Flask-WTForms

Forms

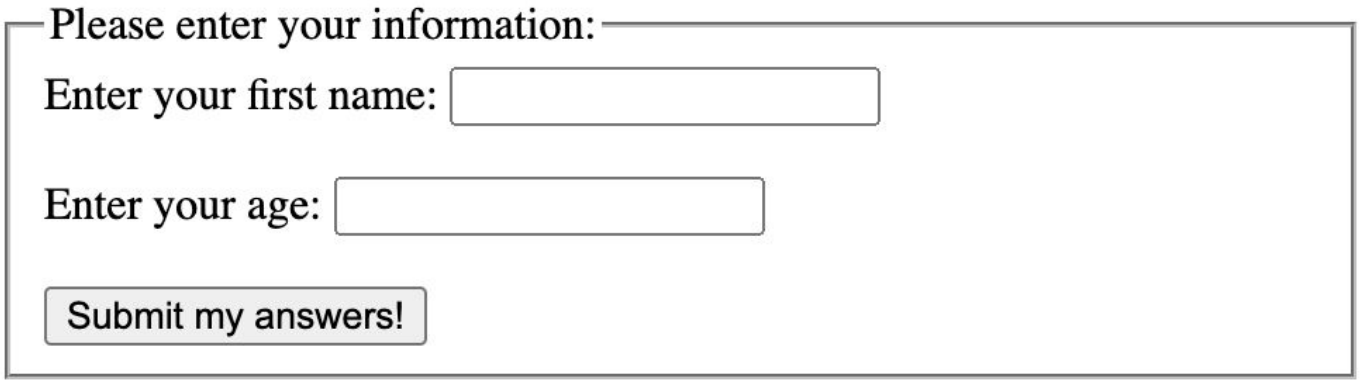
MAKE SCHOOL

Typically, a form in HTML looks something like this:

```
<form method="POST" action="/submit">
   <fieldset>
       <legend>Please enter your information:</legend>
       <label>
           Enter your first name:
           <input type="text" name="first_name">
       </label>
       <label>
           Enter your age:
           <input type="number" name="age">
       </label>
       <input type="submit" value="Submit my answers!">
   </fieldset>
</form>
```

Forms

For the user, this form would look something like this:



Forms

MAKE SCHOOL

However, what if the user enters invalid data into the form (e.g. leaves the name or age field blank)?

It can be a lot of work to validate every field in a form on the back-end, and we'd have to write a lot of repetitive code.

So, we can keep our code **D.R.Y.** by using a forms library.

Flask-WTForms

MAKE SCHOOL

WTForms is a flexible forms validation and rendering library for Python web development. Source

Flask-WTF is an integration of Flask and WTForms, and includes more features like CSRF, file upload, and reCAPTCHA. Source

These two libraries allow us to easily build out complex forms that include built-in validation.

Flask-WTForms

MAKE SCHOOL

Take a look at the Quick Start Guide for an overview.

In general, there are 3 parts to using a class-based form:

1. Write a form class containing the desired fields.
2. In the route code where you want the form displayed, create an instance of it and pass it to the template.
3. In the template code, display the form fields inside of a `<form>` element.

MAKE SCHOOL

Lab Activity

Forms Lab (25 minutes)

MAKE SCHOOL

With a partner, complete the Forms Lab activity by following the instructions for parts 1 and 2.

MAKE SCHOOL

Break - 10 min

MAKE SCHOOL

Forms Concepts

Form Fields

Typically, a form class has one **field** per form input. There are different types of fields for different input types.

If the form is used to create a model, the fields will typically correspond to the fields in the model.

```
class Book(db.Model):
    id = db.Column(db.Integer, primary_key=True)
    title = db.Column(db.String(80),
nullable=False)
    publish_date = db.Column(db.Date)
    # ...
```

```
class BookForm(FlaskForm):
    """Form to create a book."""
    title = StringField('Book Title',
        validators=[DataRequired()])
    publish_date = DateField('Date Published')
```

Validators

A validator is used to impose some constraints on the form input values.

Some examples of validators are:

- DataRequired - cannot be empty
- Email - must be an email address
- Length
- URL
- RegExp

validate_on_submit

We can create a form object within a route, and it will automatically be populated with the user's responses (if any).

If the form was submitted, **and** all values were valid, then `validate_on_submit()` will return `True`.

```
@main.route('/create_book', methods=['GET', 'POST'])
def create_book():
    form = BookForm()
    if form.validate_on_submit():
        # ... do form processing here ...
```

csrf_token

The `csrf_token` is used to prevent **Cross-Site Request Forgery (CSRF)** attacks.

We can put it in the HTML form element and it will automatically be processed on submit.

```
<form method="POST" action="/submit">
    {{ form.csrf_token }}
    <!-- ... form inputs go here ... -->
</form>
```

Lab

MAKE SCHOOL

Continue working on the lab activity.

MAKE SCHOOL

Enums

Enums

Enums (short for **enumeration**) are useful in lots of different contexts, not just forms.

Think of an enum as kind of like a boolean. For a boolean, there are only two possible values: **True** and **False**.

For an enum, there is also a finite set of possible values. But this time, we get to choose what they are.

Enums

Here is an example of an enum:

```
class DayOfWeek(enum.Enum):
    SUNDAY = 1
    MONDAY = 2
    TUESDAY = 3
    WEDNESDAY = 4
    THURSDAY = 5
    FRIDAY = 6
    SATURDAY = 7
```

This means that we can't have a day of the week that is anything other than these 7 values.

Enums

Here is another example of an enum:

```
class Audience(enum.Enum):
    CHILDREN = 'Children'
    YOUNG_ADULT = 'Young Adult'
    ADULT = 'Adult'
    ALL = 'All'
```

We can use strings to give the entries nice display values to the user.

MAKE SCHOOL

Wrap-Up

Wrap-Up

MAKE SCHOOL

Homework 2 (events site): Due on Tuesday EOD

Homework 3 (forms): Due next Thursday

Quiz 1: Available on Gradescope, due on Friday midnight

- Please reach out if you will need more time to complete it