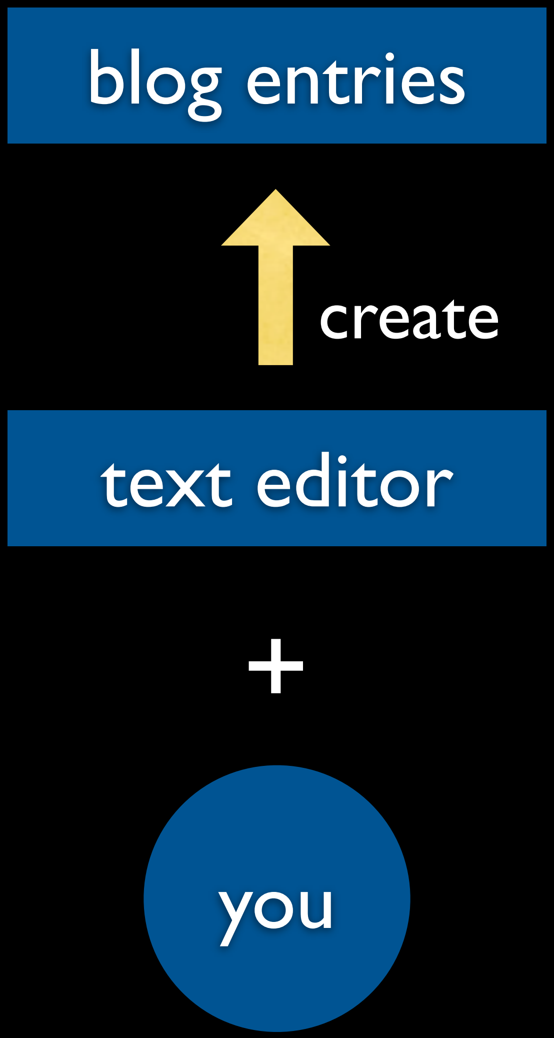
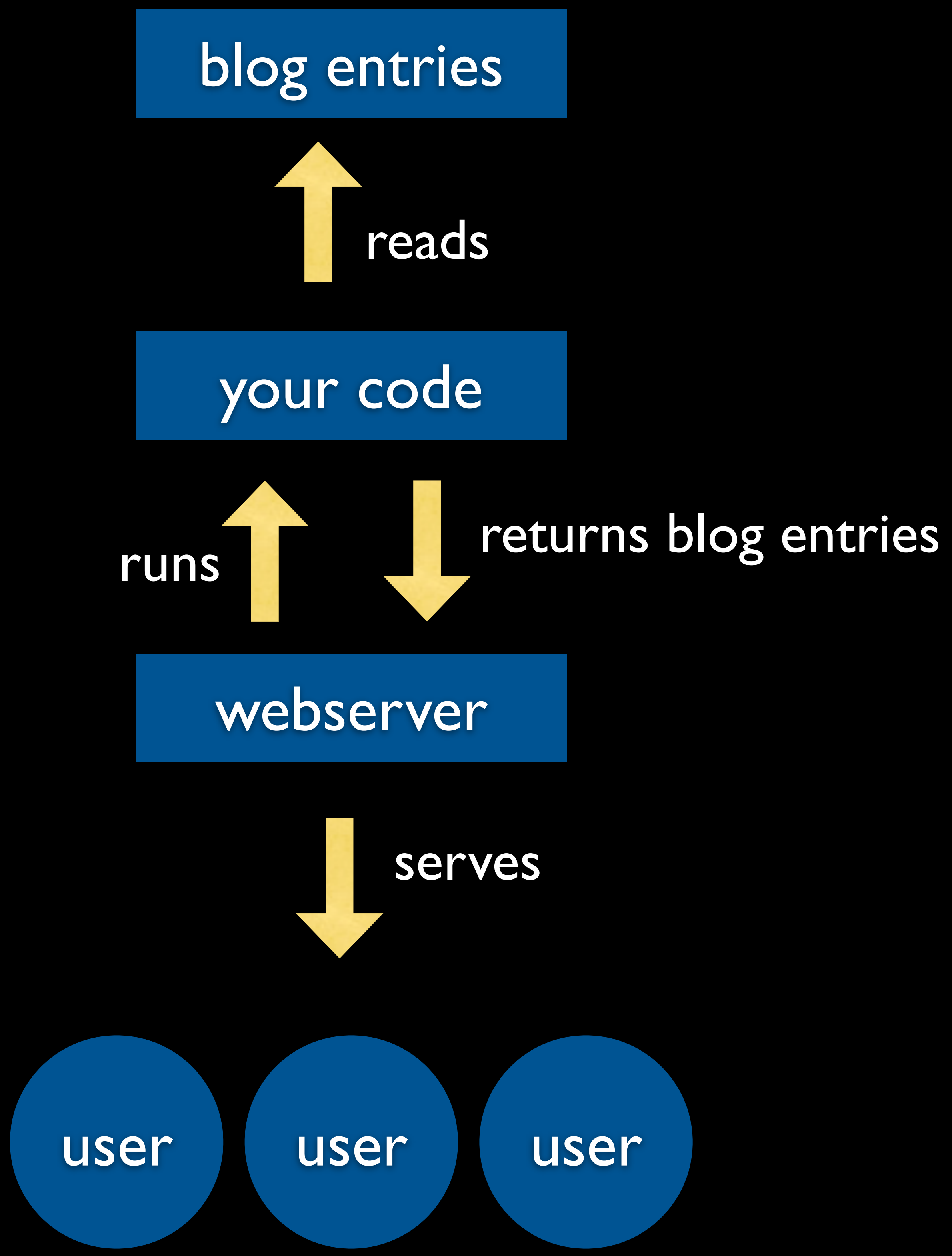
Make a blog with Flask

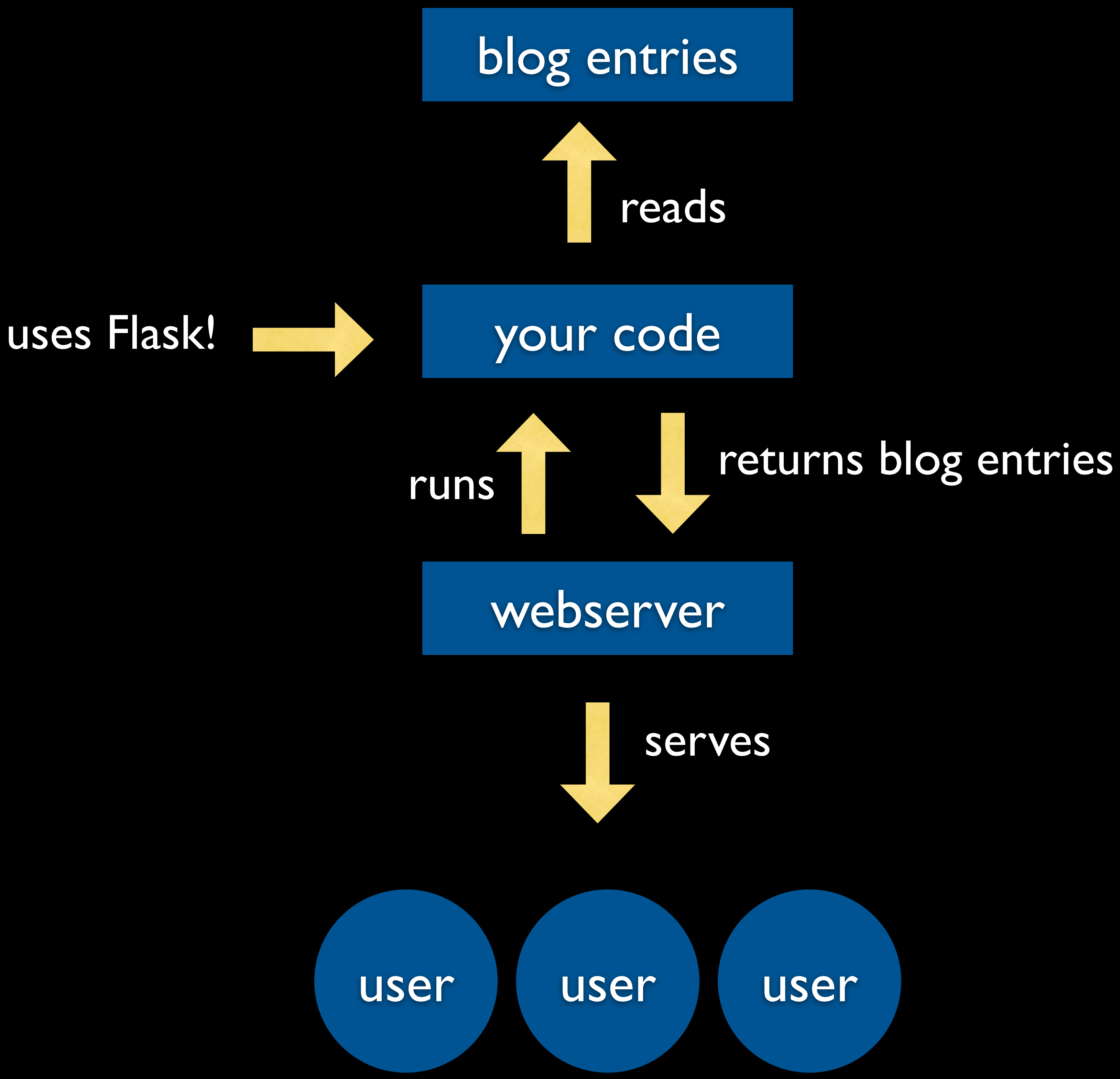
Selena Deckelmann
Data Architect, Mozilla

What is a blog?

- Text entries listed in reverse chronological order (like Twitter, but longer text. Maybe)


blog entries
create
text editor
+
you


blog entries
reads
your code
runs
returns blog entries
webserver
serves
user
user
user


blog entries
reads
uses Flask!
your code
runs
returns blog entries
webserver
serves
user
user
user

1. Open Terminal
2. Move to a directory to store your blog with `cd`

```
git clone \
 git@github.com:selenamarie/selenas-flask-blog.git

git checkout class
```

```
virtualenv --no-site-packages env

source env/bin/activate

pip install -r requirements.txt
```

Create file: `sitebuilder.py`

Add the following to `sitebuilder.py`:

```
from flask import Flask
app = Flask(__name__)

@app.route("/")
def index():
    return "Hello World!"

if __name__ == "__main__":
    app.run(port=8000)
```

Go back to Terminal and run:

```
python sitebuilder.py
```

Now open in Firefox:

```
http://127.0.0.1:8000
```

What we just did

- Created the simplest possible Flask program
- Started a webserver that ran our program
- Viewed the output from our program!

```
mkdir pages
```

Create file `pages/hello-world.md:`

```
title: Hello World
date: 2012-03-04

**Hello World**, from a *page*!
```

```
mkdir pages
```

Create file `pages/hello-world.md:`

```
title: Hello World
date: 2012-03-04

**Hello World**, from a *page*!
```

Replace `sitebuilder.py`:

```
from flask import Flask
from flaskext.flatpages import FlatPages

DEBUG = True
FLATPAGES_AUTO_RELOAD = DEBUG
FLATPAGES_EXTENSION = '.md'

app = Flask(__name__)
app.config.from_object(__name__)
pages = FlatPages(app)

@app.route('/')
def index():
    return "Hello World"
```

Add the following to `sitebuilder.py`:

(continued)

```
@app.route('/<path:path>/')
def page(path):
    return pages.get_or_404(path).html

if __name__ == '__main__':
    app.run(port=8000)
```

Go back to Terminal and run:

```
python sitebuilder.py
```

Now open in Firefox:

```
http://127.0.0.1:8000
```

What we just did

- Created a Markdown-formatted text file
- Created a program that reads Markdown files
- Ran our program in a webserver
- Viewed the output from our program

```
mkdir templates
```

Create file `templates/base.html`:

```
<!doctype html>
<html>
<head>
    <meta charset="utf-8">
    <title>My site</title>
</head>
<body>
    <h1><a href="{{ url_for("index") }}">My site</a></h1>
{% block content %}
    <p>Default content to be displayed</p>
{% endblock content %}
</body>
</html>
```

```
mkdir templates
```

Create file `templates/page.html:`

```
{% extends "base.html" %}

{% block content %}
    <h2>{{ page.title }}</h2>
    {{ page.html|safe }}
{% endblock content %}
```

Update `sitebuilder.py`:

```
@app.route('/<path:path>/')
def page(path):
    page = pages.get_or_404(path)
    return render_template('page.html', page=page)
```

Go back to Terminal and run:

```
python sitebuilder.py
```

Now open in Firefox:

```
http://127.0.0.1:8000
```

What we just did

- Created a template
- Added code to interpret the template
- Ran our program in a webserver
- Viewed the output from our program

Create file `templates/index.html:`

```
{% extends "base.html" %}

{% block content %}
    <h2>List of stuff</h2>
    <ul>
    {% for page in pages %}
        <li>
            <a href="{{ url_for("page",
            path=page.path) }}">{{ page.title }}</a>
        </li>
    {% else %}
        <li>No stuff.</li>
    {% endfor %}
    </ul>
{% endblock content %}
```

Update `sitebuilder.py`:

```
@app.route('/')
def index():
    return render_template('index.html',
       pages=pages)
```

Go back to Terminal and run:

```
python sitebuilder.py
```

Now open in Firefox:

```
http://127.0.0.1:8000
```

What we just did

- Created another template for “index”
- Added code to interpret the template
- Ran our program in a webserver
- Viewed the output from our program

Update file `pages/hello-world.md:`

```
title: Hello World
date: 2012-03-04
tags: [general, awesome, stuff]

**Hello World**, from a *page*!
```

Update file `templates/index.html`:

```
{% extends "base.html" %}

{% block content %}
    <h2>List of stuff</h2>
    {% with pages=pages  %}
        {% include "_list.html" %}
    {% endwith %}
{% endblock content %}
```

Create file `templates/_list.html`:

```
<ul>
{% for page in pages %}
    <li>
        <a href="{{ url_for("page",
path=page.path) }}">{{ page.title }}</a>
    {% if page.meta.tags|length %}
        | Tagged:
        {% for page_tag in page.meta.tags %}
            <a href="{{ url_for("tag",
            tag=page_tag) }}">{{ page_tag }}</a>
        {% endfor %}
    {% endif %}
    </li>
{% else %}
    <li>No page.</li>
{% endfor %}
</ul>
```

Add this to `sitebuilder.py`:

```
@app.route('/tag/<string:tag>/')
def tag(tag):
    tagged = [p for p in pages if tag in p.meta.get('tags', [])]
    return render_template('tag.html', pages=tagged, tag=tag)
```

Go back to Terminal and run:

```
python sitebuilder.py
```

Now open in Firefox:

```
http://127.0.0.1:8000
```

What we just did

- Added support for “tag” metadata
- Added code to interpret the metadata
- Added templates to use the metadata
- Ran our program in a webserver
- Viewed the output from our program