Static Site Generators

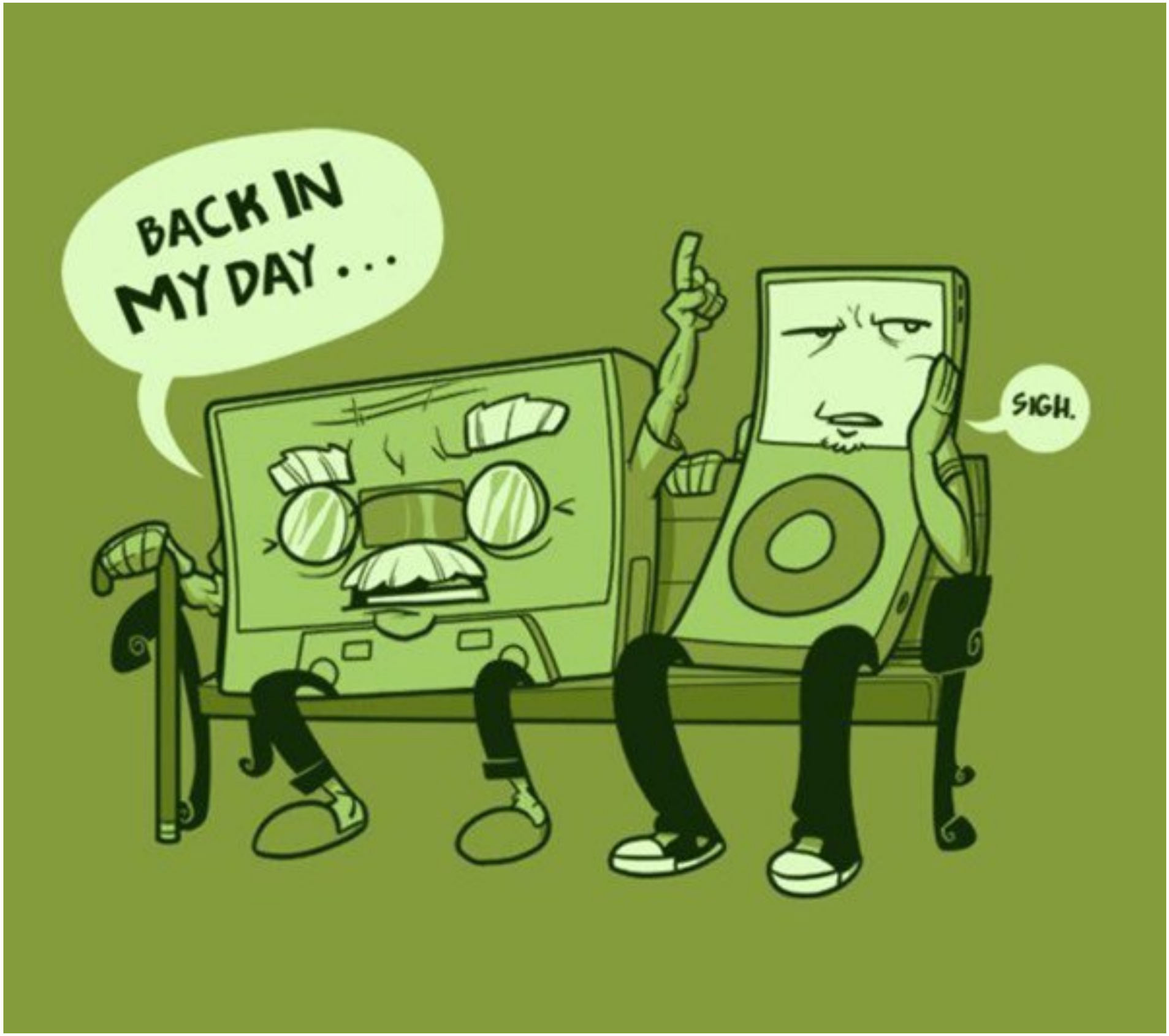
Greg Schafer
25 Feb 2016

Outline

- History
- Purpose
- Pros/Cons
- Examples
- Let’s try out Lektor!

Outline

- History
- Purpose
- Pros/Cons
- Examples
- Let’s try out Lektor!


BACK IN MY DAY . . .
SIGH.

YAHOO! CELEBRATE AND WIN. HOLIDAY EXTRAVAGANZA CLICK HERE.

Get Local | Weekly Picks

Search | Options

Yellow Pages - People Search - City Maps -- Stock Quotes - Sports Scores

- **Arts and Humanities** - Architecture, Photography, Literature...
- **Business and Economy [Xtra!]** - Companies, Investments, Classifieds...
- **Computers and Internet [Xtra!]** - Internet, WWW, Software, Multimedia...
- **Education** - Universities, K-12, College Entrance...
- **Entertainment [Xtra!]** - Cool Links, Movies, Music, Humor...
- **Government** - 96 Elections, Politics [Xtra!], Agencies, Law, Military...
- **Health [Xtra!]** - Medicine, Drugs, Diseases, Fitness...
- **News and Media [Xtra!]** - Current Events, Magazines, TV, Newspapers...
- **Recreation and Sports [Xtra!]** - Sports, Games, Travel, Autos, Outdoors...
- **Reference** - Libraries, Dictionaries, Phone Numbers...
- **Regional** - Countries, Regions, U.S. States...
- **Science** - CS, Biology, Astronomy, Engineering...

STARWAVE - ESPNET SPORTSZONE • NBA.COM • NASCAR ONLINE • OUTSIDE ONLINE • MR. SHOWBIZ • FAMILY PLANET

Family Planet

A SERVICE OF STARWAVE

Failure: Loss of Face or

Learning Experience?

What's new on the Planet for THURSDAY, OCTOBER 17, 1996

Today! **NOW! Talk live with party expert Caryl Krueger!**

HEADLINES

- U.S. kids poorest in the industrialized world
- Three-step licenses to make teens drive safer
- Juvenile drug crime rises

MORE

- great stuff on the table of contents

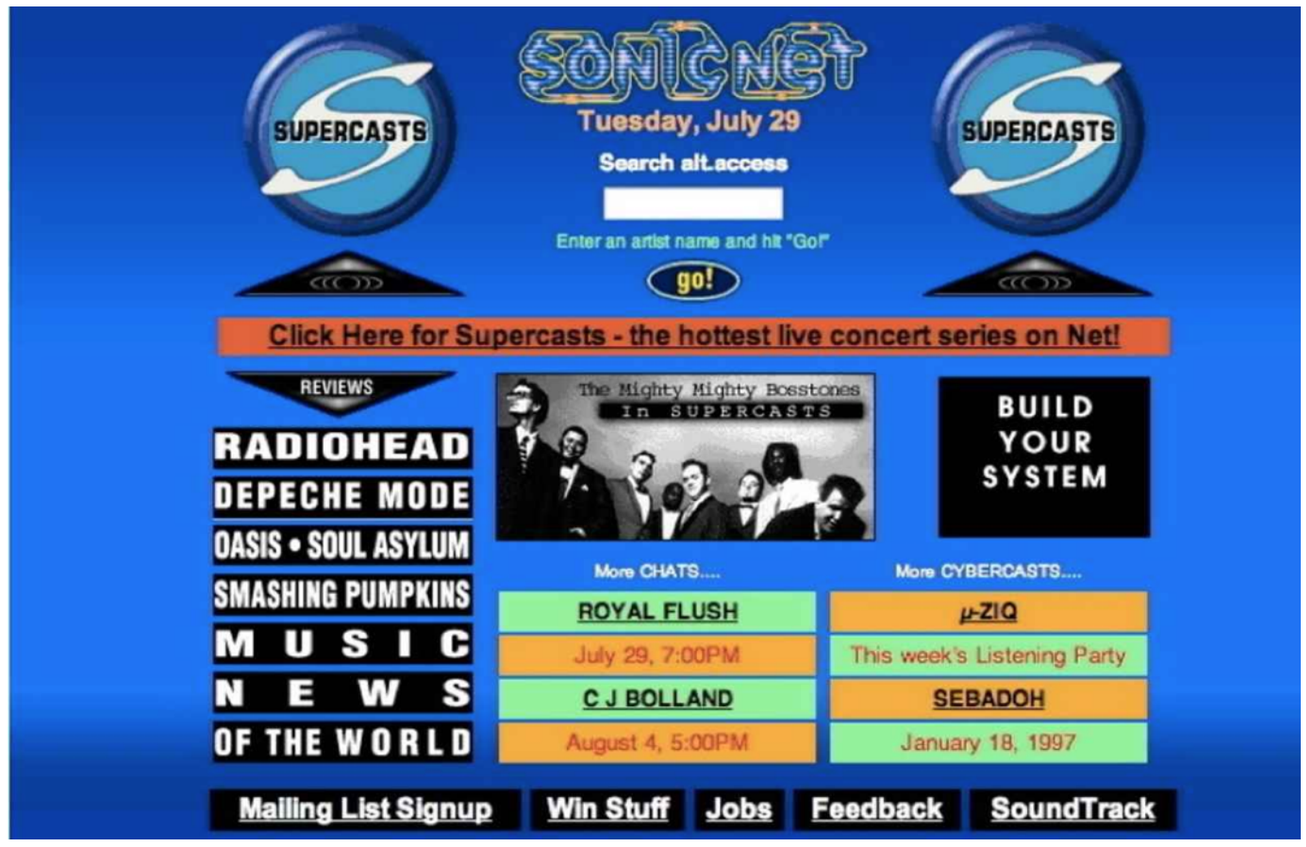
ELECTION WATCH

HALLOWEEN

- Your one-stop spot for all things ghostly
- Read this before you touch that pumpkin!
- Tell us a tale of a Halloween past

FUN!

CONTENTS | NEWS | FUN | SOUND OFF



Tuesday, August 15, 2000

epicurious

FOR PEOPLE WHO EAT

EATING | DRINKING | PLAYING

epicurious TV | BON APPÉTIT | Gourmet

GRAND OPENING
Visit our new specialty food store

IN SEASON

BASKET CASE:
Sane ideas and menus for stress-free picnics

SWEEPSTAKES

Win a gourmet Dean & DeLuca picnic basket from Epicurious!

TODAY

DRINK: Iced Coffee Fillip
RECIPE: Spiced Pork Tenderloin
STORE: S'mores Collection
TIP: A Use for Onion and Garlic Skins

RECIPE FILE

HUNGRY? Search more than 11,000 recipes

GO

- More recipe search options
- Most talked about recipes at Constant Comment
- Search Spy: See what Epicureans crave now

Go to your RECIPE BOX

SEE FOOD

FEATURED SLIDESHOWS

Desserts
Grilling
Ice Cream & Sorbet
Salads
Side Dishes
Summer Soups

NEW BATCH DISPATCH
Our latest recipes delivered to your inbox weekly

SIGN UP

SHOP EPICURIOUS

SAVOR SUMMER

Visit our new specialty food store for a taste of the season's best

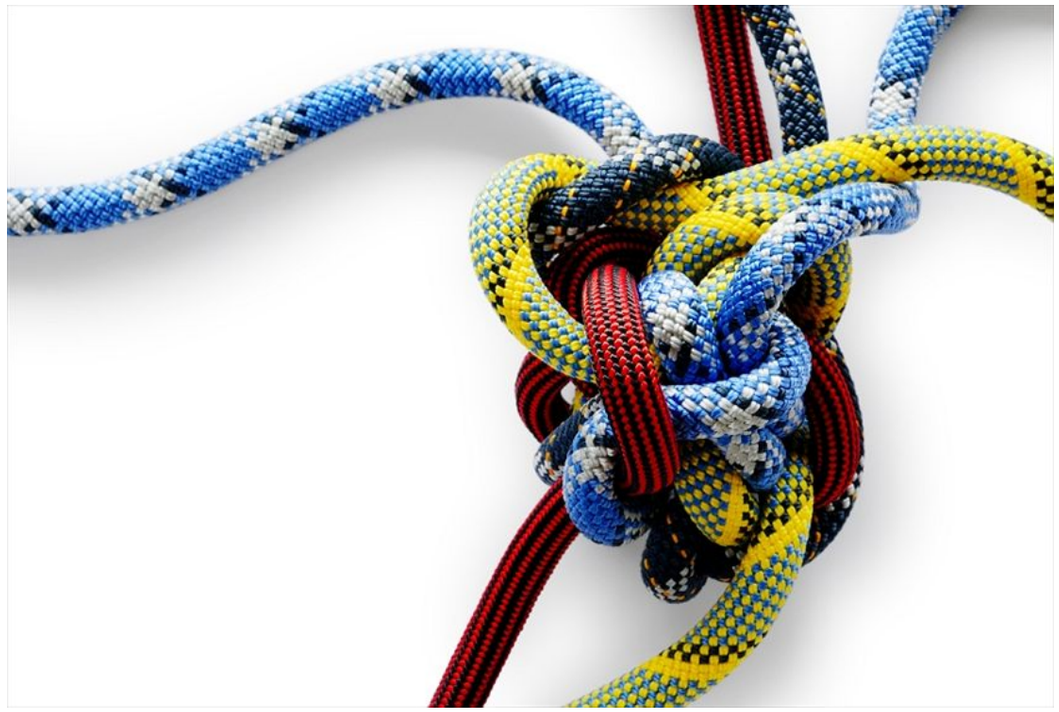
Don't miss...Epson two person picnic basket

Challenges of dynamic

- Complexity
- Security (i.e. WordPress)
- Speed
  - Caching
- Traffic spikes

Challenges of dynamic

- Complexity
- Security (i.e. WordPress)
- Speed
  - Caching
- Traffic spikes



Challenges of dynamic

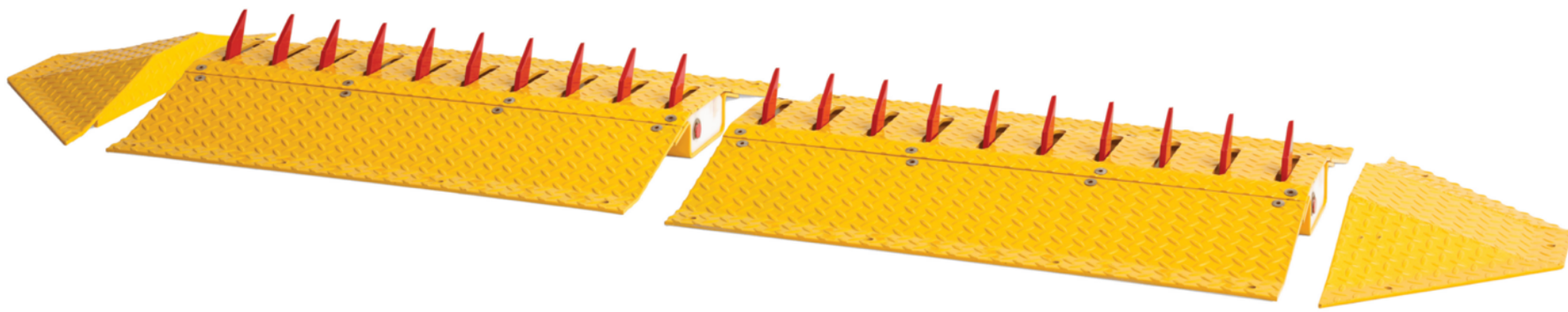
- Complexity
- Security (i.e. WordPress)
- Speed
  - Caching
- Traffic spikes

Challenges of dynamic

- Complexity
- Security (i.e. WordPress
- Speed
  - Caching
- Traffic spikes

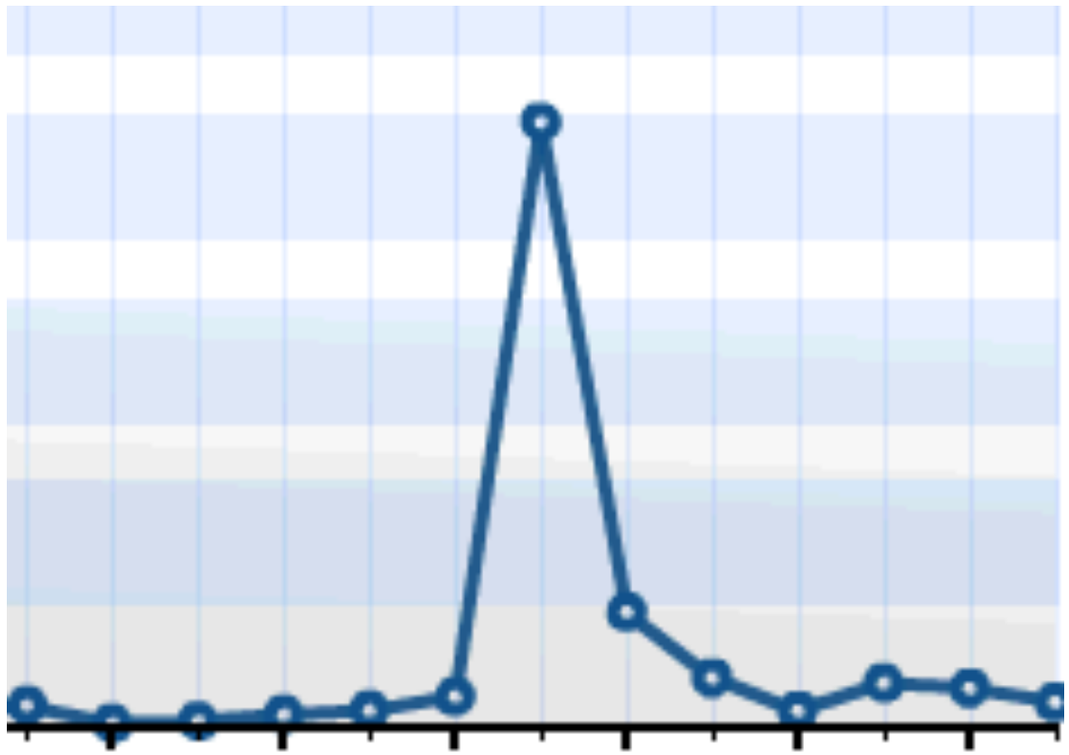
Challenges of dynamic

- Complexity
- Security (i.e. WordPress)
- Speed
  - Caching
- Traffic spikes



Challenges of dynamic

- Complexity
- Security (i.e. WordPress)
- Speed
  - Caching
- Traffic spikes

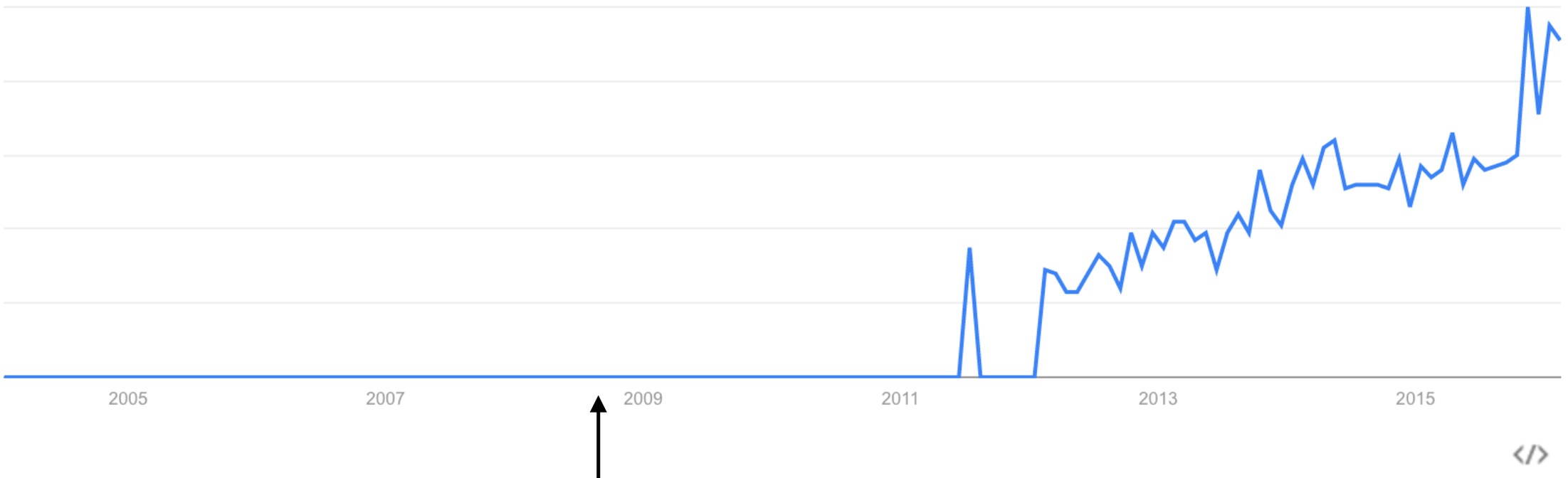


static site generator
Search term

+ Add term

Interest over time

News headlines Forecast



Jekyll announced

Static Site Generators

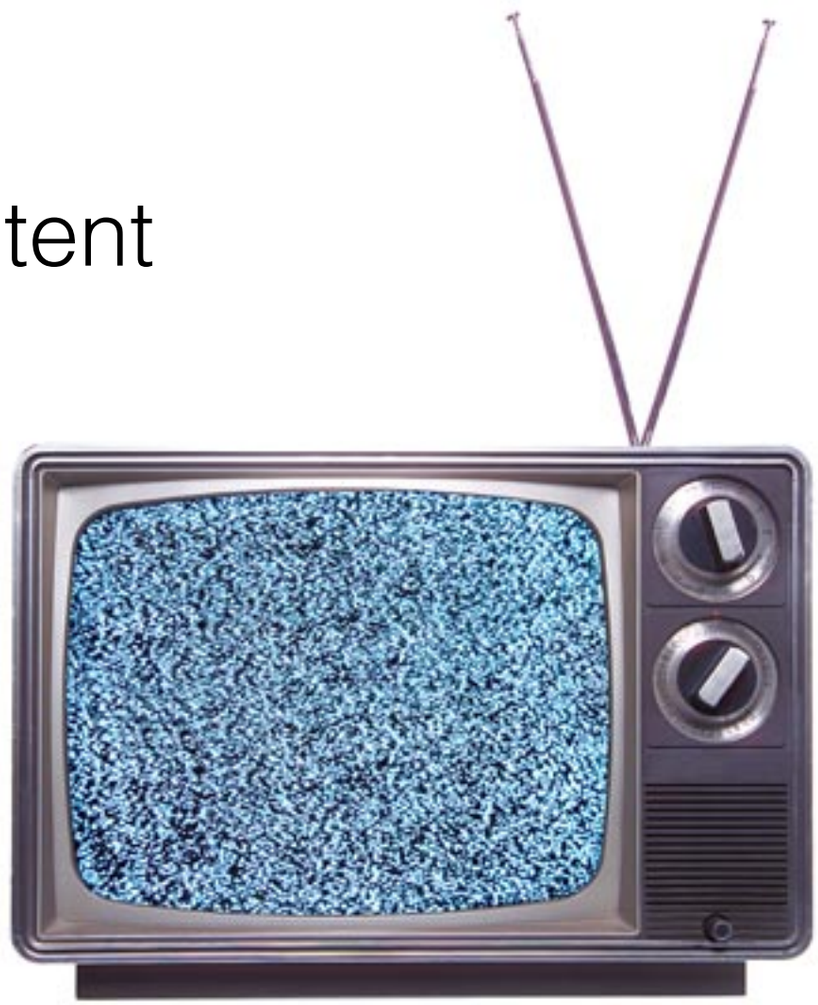
- Produce HTML pages
- Features
  - Templating
  - Markdown support
  - Metadata or content model
  - Asset pipeline

Advantages

- Speed
- Version control for content
- Security
- Simpler deployment, maintenance
- Scalable (traffic spikes)

Disadvantages

- They’re uhh… static
    - No real-time or customized content
    - No user input
- Authoring tools lacking



Generator	Released	Language	Template	Notable for	Used by
Jekyll	2008	Ruby	Liquid	Maturity, support	GitHub Pages, healthcare.gov, tons more
Middleman	2008	Ruby	Ruby ERB templates	Broader capabilities	Mailchimp, Nest, Simple
Roots	2012	Javascript	Jade	Extensible, mature asset pipeline	Carrot, Vice Media group
Hugo	2013	Go	Amber, Ace	Fast, but a bit barebones	UCSB, Gophercon
Pelican	2010	Python	Jinja2	Most popular python generator	Linux Kernel Archives
Lektor	2015	Python	Jinja2	Includes admin	Not much yet

Lektor

- Armin Ronacher
  - Flask, Jinja2, Werkzeug, other Pocoo
- Python
- Released in Dec 2015
- Motivation: “run a simple website in a secure manner without having to resort to all kinds of user-unfriendly hackery"

```
lektor-sites$ lektor quickstart
Lektor Quickstart
==================

This wizard will generate a new basic project with some sensible defaults for
getting started quickly.  We jsut need to go through a few questions so that
the project is set up correctly for you.

Step 1:
| A project needs a name.  The name is primarily used for the admin UI and
| some other places to refer to your project to not get confused if multiple
| projects exist.  You can change this at any later point.
> Project Name: pythonweb-demo

Step 2:
| This is the path where the project will be located.  You can move a
| project around later if you do not like the path.  If you provide a
| relative path it will be relative to the working directory.
> Project Path [/Users/occipital/repos/lektor-sites/pythonweb-demo]:

Step 3:
| Do you want to generate a basic blog module?  If you enable this the
| models for a very basic blog will be generated.
> Add Basic Blog [Y/n]: Y

Step 4:
| Your name.  This is used in a few places in the default template to refer
| to in the default copyright messages.
> Author Name [greg]: Greg Schafer

That's all. Create project? [Y/n] Y
```

.lektorproject

- name
- locales
- deployment settings
- external packages

```
pythonweb-demo$ tree
.
├── assets
|   └── static
|       └── style.css
├── content
|   ├── about
|   |   └── contents.lr
|   ├── blog
|   |   ├── contents.lr
|   |   └── first-post
|   |       └── contents.lr
|   ├── contents.lr
|   └── projects
|       └── contents.lr
├── models
|   ├── blog-post.ini
|   ├── blog.ini
|   └── page.ini
├── pythonweb-demo.lektorproject
└── templates
    ├── blog-post.html
    ├── blog.html
    ├── layout.html
    ├── macros
    |   ├── blog.html
    |   └── pagination.html
    └── page.html

10 directories, 16 files
```

Content

- 1 folder —> 1 page —> 1 URL
  - “about” folder —> /about
- contents.lr contains the content

```
pythonweb-demo$ tree
.
├── assets
|   └── static
|       └── style.css
├── content
|   ├── about
|   |   └── contents.lr
|   ├── blog
|   |   ├── contents.lr
|   |   └── first-post
|   |       └── contents.lr
|   ├── contents.lr
|   └── projects
|       └── contents.lr
├── models
|   ├── blog-post.ini
|   ├── blog.ini
|   └── page.ini
├── pythonweb-demo.lektorproject
└── templates
    ├── blog-post.html
    ├── blog.html
    ├── layout.html
    ├── macros
    |   ├── blog.html
    |   └── pagination.html
    └── page.html

10 directories, 16 files
```

```
pythonweb-demo$ cat content/about/contents.lr
title: About this Website
---
body:

This is a website that was made with the Lektor quickstart.

And it does not contain a lot of information.
```


About this Website

This is a website that was made with the Lektor quickstart.

And it does not contain a lot of information.

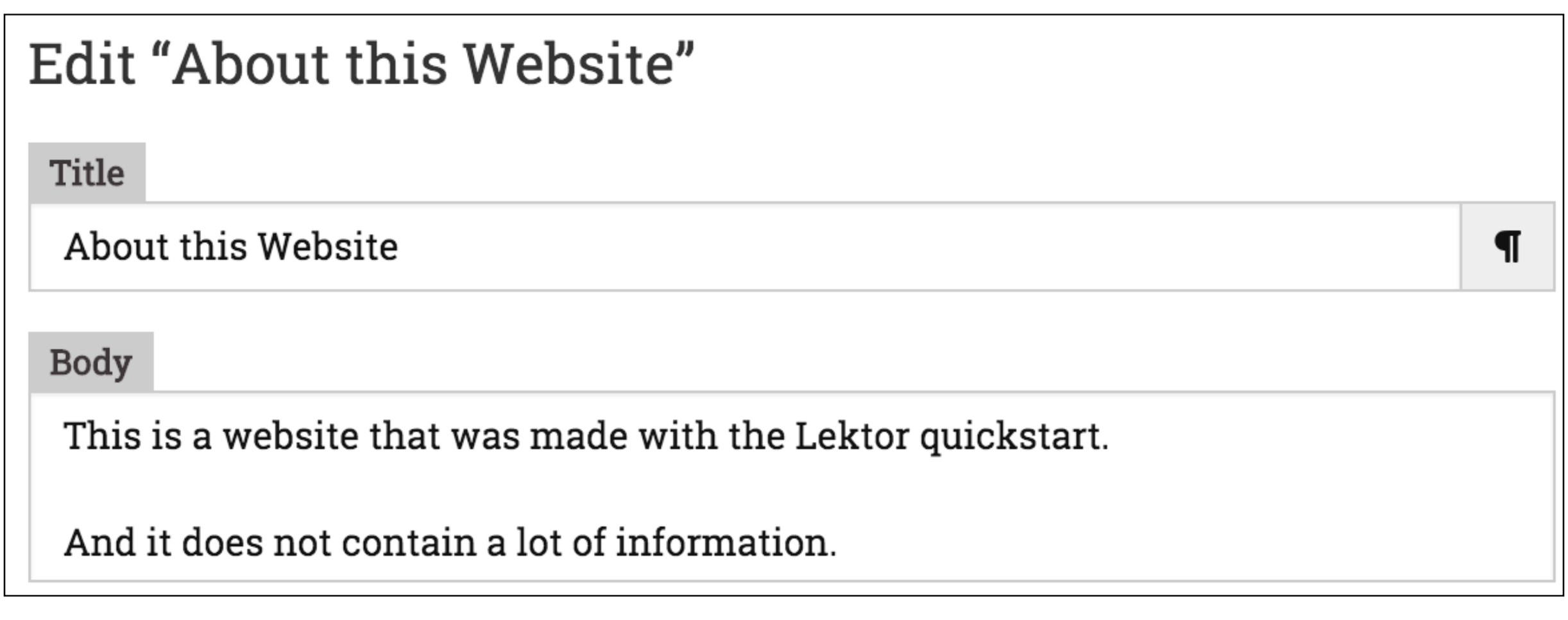
Models

- Define which fields exist and what goes in them
- Default model is “page”

```
pythonweb-demo$ tree
.
├── assets
|   └── static
|       └── style.css
├── content
|   ├── about
|   |   └── contents.lr
|   ├── blog
|   |   ├── contents.lr
|   |   └── first-post
|   |       └── contents.lr
|   ├── contents.lr
|   └── projects
|       └── contents.lr
├── models
|   ├── blog-post.ini
|   ├── blog.ini
|   └── page.ini
├── pythonweb-demo.lektorproject
└── templates
    ├── blog-post.html
    ├── blog.html
    ├── layout.html
    ├── macros
    |   ├── blog.html
    |   └── pagination.html
    └── page.html

10 directories, 16 files
```

models/page.ini

```ini
[model]
name = Page
label = {{ this.title }}

[fields.title]
label = Title
type = string

[fields.body]
label = Body
type = markdown
```



Templates

- Defines how to render the corresponding model as html
  - “page” model rendered by “page.html” template
- Jinja2

```
pythonweb-demo$ tree
.
├── assets
|   └── static
|       └── style.css
├── content
|   ├── about
|   |   └── contents.lr
|   ├── blog
|   |   ├── contents.lr
|   |   └── first-post
|   |       └── contents.lr
|   ├── contents.lr
|   └── projects
|       └── contents.lr
├── models
|   ├── blog-post.ini
|   ├── blog.ini
|   └── page.ini
├── pythonweb-demo.lektorproject
└── templates
    ├── blog-post.html
    ├── blog.html
    ├── layout.html
    ├── macros
    |   ├── blog.html
    |   └── pagination.html
    └── page.html

10 directories, 16 files
```

templates/page.html

```
1 {% extends "layout.html" %}
2 {% block title %}{{ this.title }}{% endblock %}
3 {% block body %}
4   <h2>{{ this.title }}</h2>
5   {{ this.body }}
6 {% endblock %}
```

templates/layout.html

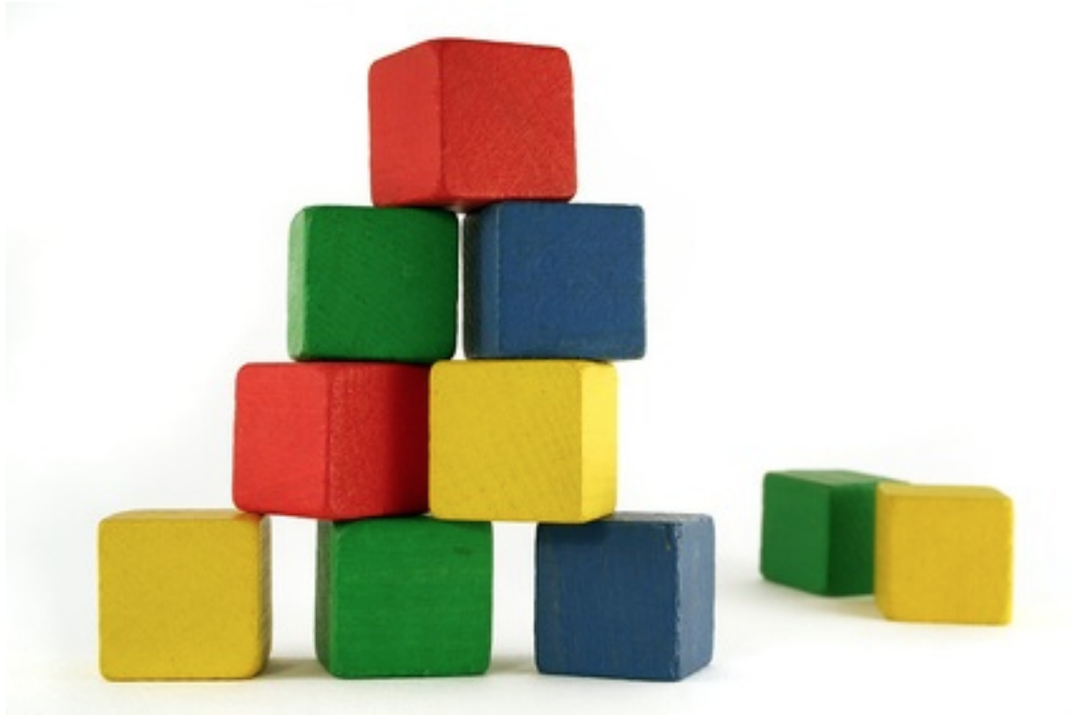
```
<!doctype html>
<meta charset="utf-8">
<link rel="stylesheet" href="{{ '/static/style.css'|url }}">
<title>{% block title %}Welcome{% endblock %} - pythonweb-demo</title>
<body>
  <header>
    <h1>pythonweb-demo</h1>
    <nav>
      <ul class="nav navbar-nav">
        <li{% if this._path == '/' %} class="active"{% endif %}>
          <a href="{{ '/'|url }}">Welcome</a>
        </li>
        {% for href, title in [
          ['/blog', 'Blog'],
          ['/projects', 'Projects'],
          ['/about', 'About']
        ] %}
          <li{% if this.is_child_of(href) %} class="active"{% endif %}>
            <a href="{{ href|url }}">{{ title }}</a>
          </li>
        {% endfor %}
      </ul>
    </nav>
  </header>
  <div class="page">
    {% block body %}{% endblock %}
  </div>
  <footer>
    &copy; Copyright 2016 by Greg Schafer.
  </footer>
</body>
```

Assets

- Copied as-is to output
- css, javascript, favicon.ico, .htaccess, etc.
- Can use plugins (e.g. webpack) to render less/sass to assets folder

```
pythonweb-demo$ tree
.
├── assets
│   └── static
│       └── style.css
├── content
│   ├── about
│   │   └── contents.lr
│   ├── blog
│   │   ├── contents.lr
│   │   └── first-post
│   │       └── contents.lr
│   ├── contents.lr
│   └── projects
│       └── contents.lr
├── models
│   ├── blog-post.ini
│   ├── blog.ini
│   └── page.ini
├── pythonweb-demo.lektorproject
└── templates
    ├── blog-post.html
    ├── blog.html
    ├── layout.html
    ├── macros
    │   ├── blog.html
    │   └── pagination.html
    └── page.html

10 directories, 16 files
```

Building on the Basics



People make a living doing this!

(aka adding a blog post)

Writing is hard

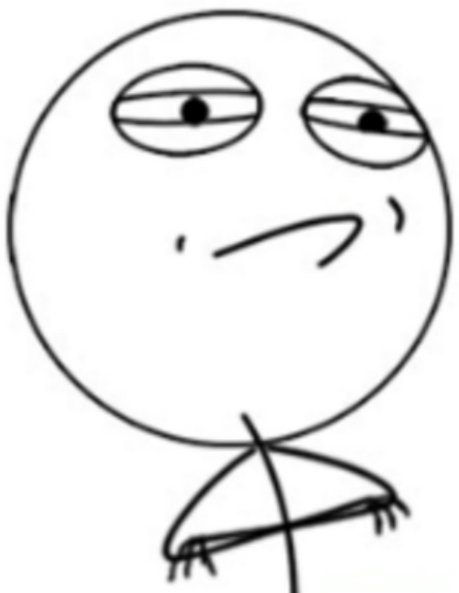
(aka adding pictures instead)

Tag all the things

(aka adding categories)

Laziness is a virtue

(aka automating navbar content)



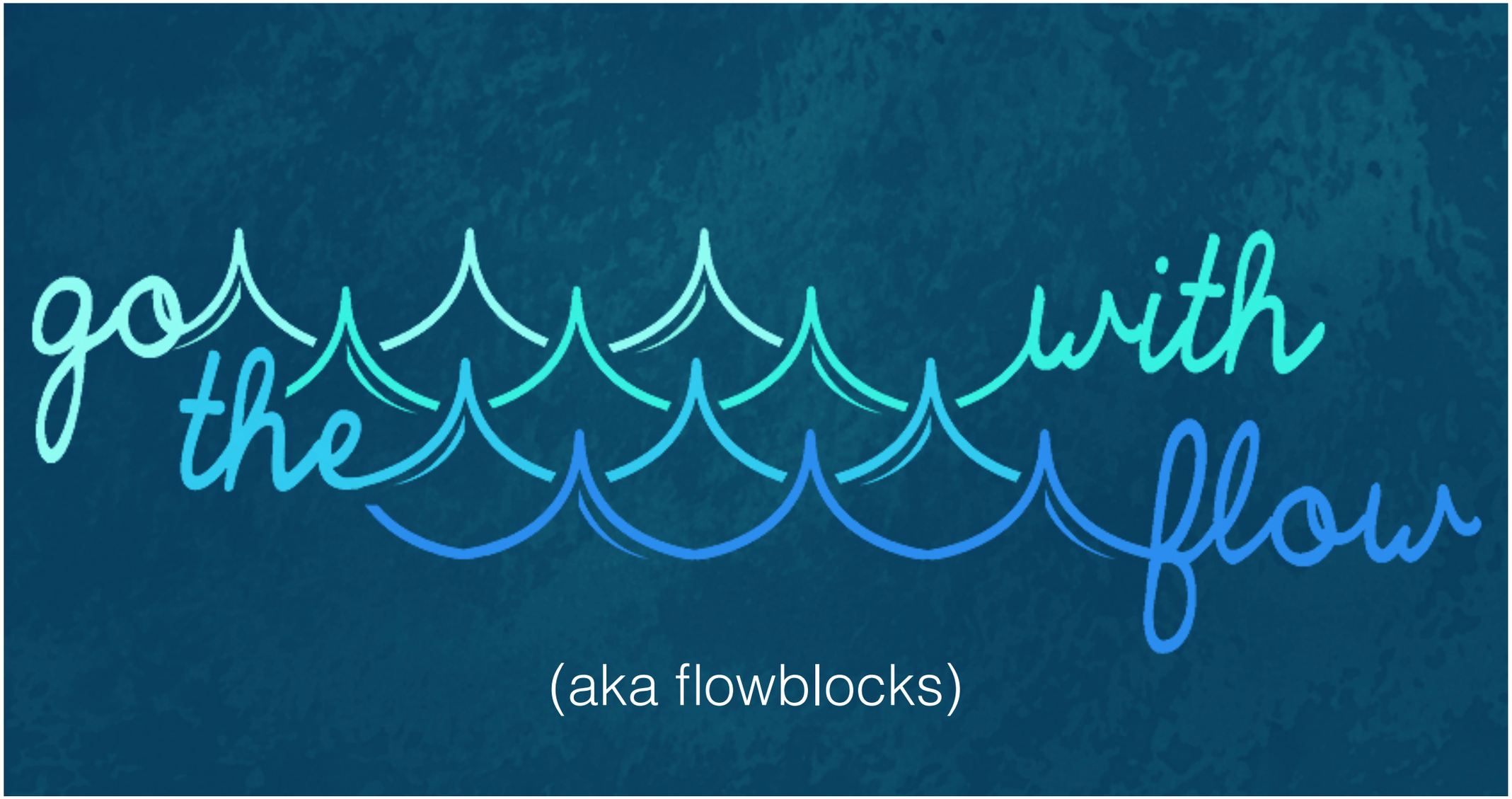
Laziness is a virtue

(aka automating navbar content)

Databags

```
[download]
path = /downloads
label = Download

[docs]
path = /docs
label = Documentation

[blog]
path = /blog
label = Blog
~
databags/menu.ini
```

```
      <div id="navbar" class="collapse navbar-collapse">
        <ul class="nav navbar-nav">
          {% for id, item in bag('menu').iteritems() %}
            <li{% if this.is_child_of(item.path) %} class="active"{% endif
              %}><a href="{{ item.path|url }}">{{ item.label }}</a></li>
          {% endfor %}
        </ul>
      </div>
    </div>
templates/layout.html                                          19,9
```


go with the flow
(aka flowblocks)

lektor-website is a good reference

but rewind to eaeaec7 or so to avoid lektor 2.0 errors

Thanks!

Questions?

Thanks!

Questions?