adaptTo()

APACHE SLING & FRIENDS TECH MEETUP

BERLIN, 28-30 SEPTEMBER 2015

Modern Web Applications with Sightly

@raducotescu
Apache Sling committer
Computer Scientist @ Adobe Systems
radu@apache.org

Adobe

rev 2.20151005

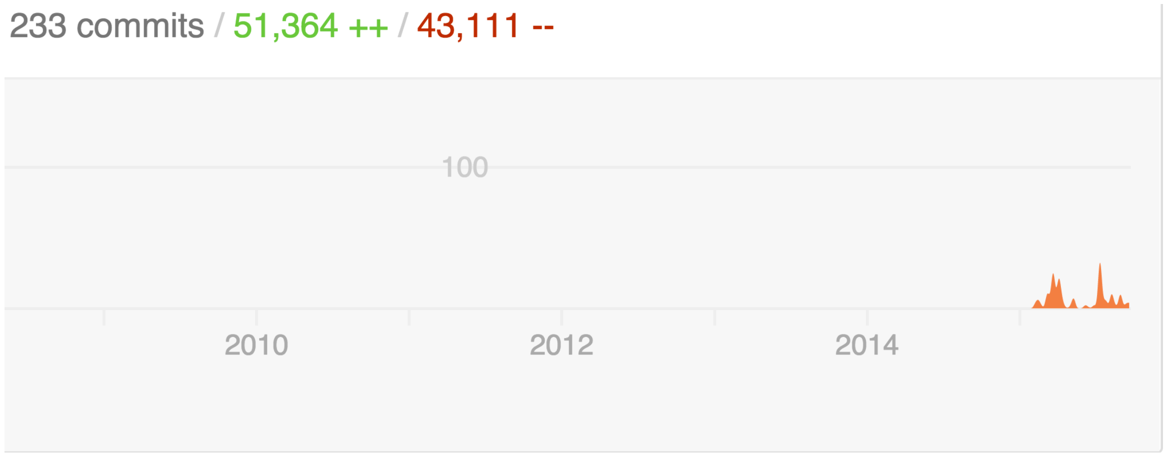
Modern Web Applications with Sightly

We’re talking about Sightly? Again?!

Modern Web Applications with Sightly

What's changed since last September?

1. The Sightly reference implementation is now a part of Apache Sling's core bundles and it's actively maintained



Modern Web Applications with Sightly

What’s changed since last September?

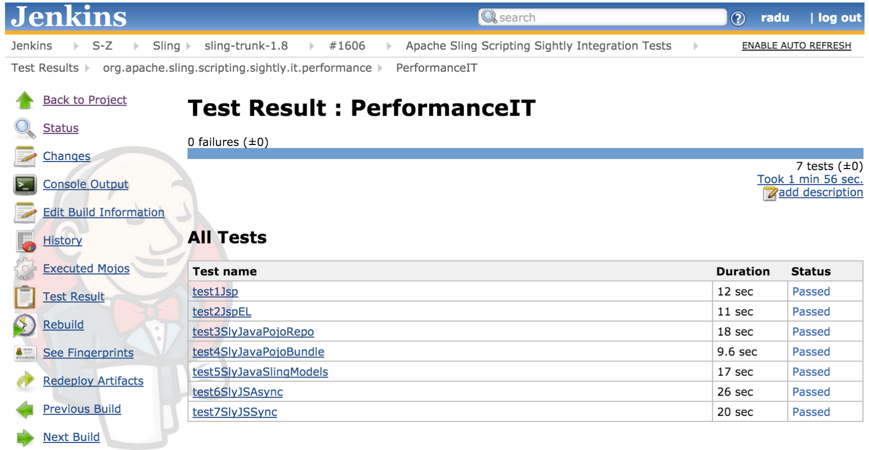
2. In January 2015 version 1.1 of the language’s specification was published:

    - no more reserved option names
    - URI manipulation options
    - `data-sly-repeat` (similar to `data-sly-list`)
    - `<sly/>` (simpler alternative to `data-sly-unwrap`)

Modern Web Applications with Sightly

What's changed since last September?

3. Several performance improvements brought Sightly on-par with the JSP Scripting Engine (backed by performance tests)



Jenkins | search | radu | log out

Jenkins ▸ S-Z ▸ Sling ▸ sling-trunk-1.8 ▸ #1606 ▸ Apache Sling Scripting Sightly Integration Tests ▸ ENABLE AUTO REFRESH

Test Results ▸ org.apache.sling.scripting.sightly.it.performance ▸ PerformanceIT

Test Result : PerformanceIT

0 failures (±0)

7 tests (±0)
Took 1 min 56 sec.
add description

All Tests

Test name	Duration	Status
test1Jsp	12 sec	Passed
test2JspEL	11 sec	Passed
test3SlyJavaPojoRepo	18 sec	Passed
test4SlyJavaPojoBundle	9.6 sec	Passed
test5SlyJavaSlingModels	17 sec	Passed
test6SlyJSAsync	26 sec	Passed
test7SlyJSSync	20 sec	Passed

Modern Web Applications with Sightly

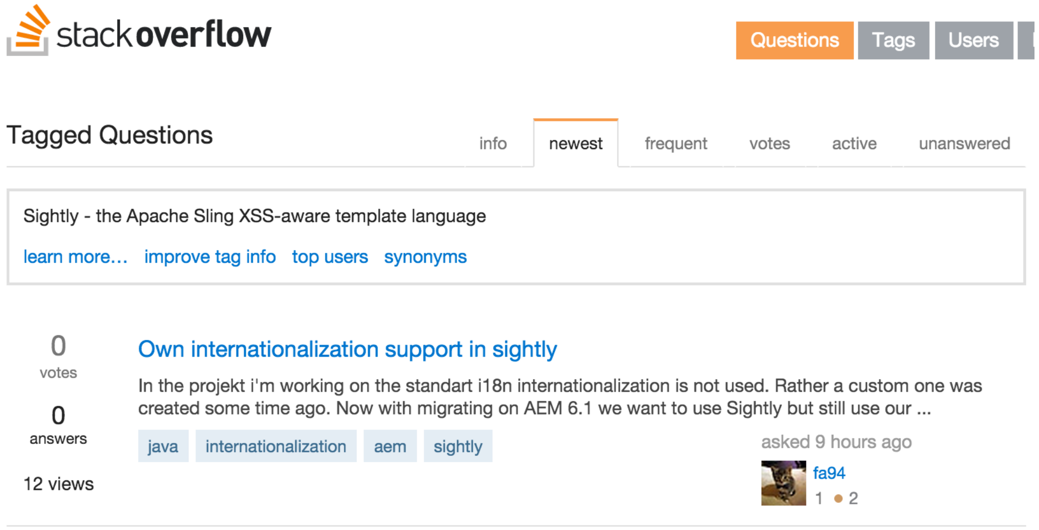
What's changed since last September?

4. We've deprecated the asynchronous JavaScript Use API (mostly a clone of the Resource API) in favour of a leaner synchronous API provided by the `org.apache.sling.scripting.javascript` bundle.

Modern Web Applications with Sightly

What's changed since last September?

5. More developers have started using Sightly in their projects



Tagged Questions

info | newest | frequent | votes | active | unanswered

Sightly - the Apache Sling XSS-aware template language

learn more… improve tag info top users synonyms

0 votes

0 answers

12 views

Own internationalization support in sightly

In the projekt i'm working on the standart i18n internationalization is not used. Rather a custom one was created some time ago. Now with migrating on AEM 6.1 we want to use Sightly but still use our ...

java internationalization aem sightly

asked 9 hours ago

fa94

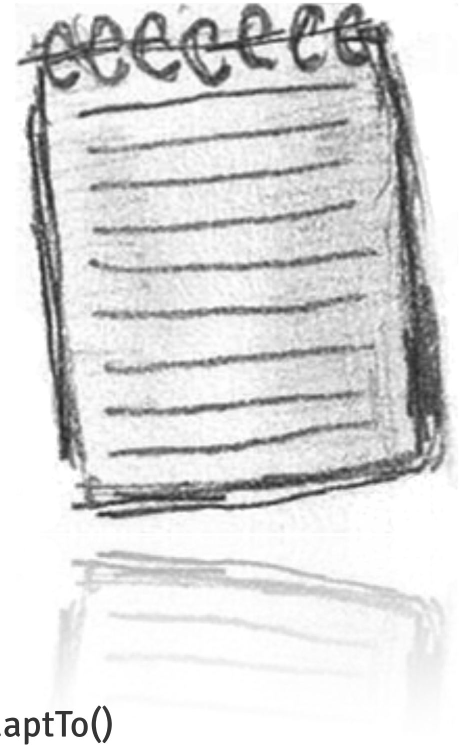
1 ● 2

Modern Web Applications with Sightly

We have a language specification. We also have a reference implementation. But I still think we're missing something…

Modern Web Applications with Sightly

Sightly Best Practices



1. Components content structure
   Organising our components thinking about reusability and flexibility is really important.

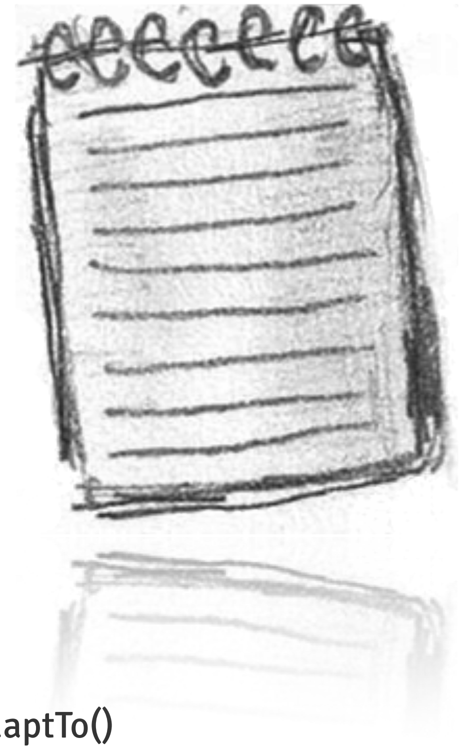
2. Markup
   We need to use the simplest markup that does the job.

3. Use API
   Having too many options can be confusing at times.

adaptTo()

Modern Web Applications with Sightly

Components content structure



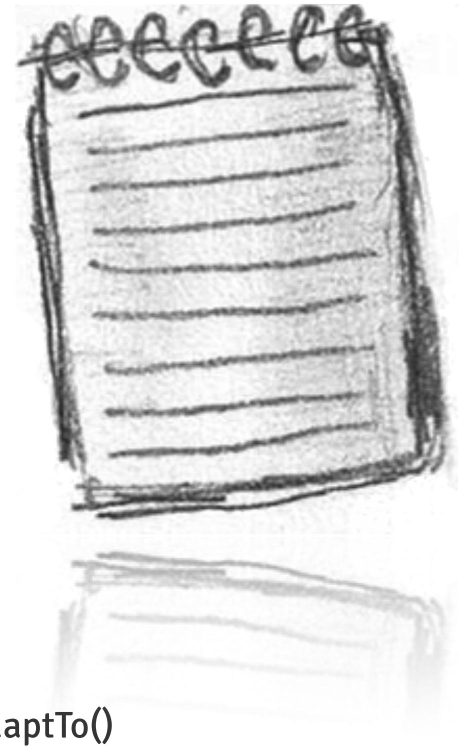
1. Try to define a structure that's flexible enough that will naturally guide you to inherit from it

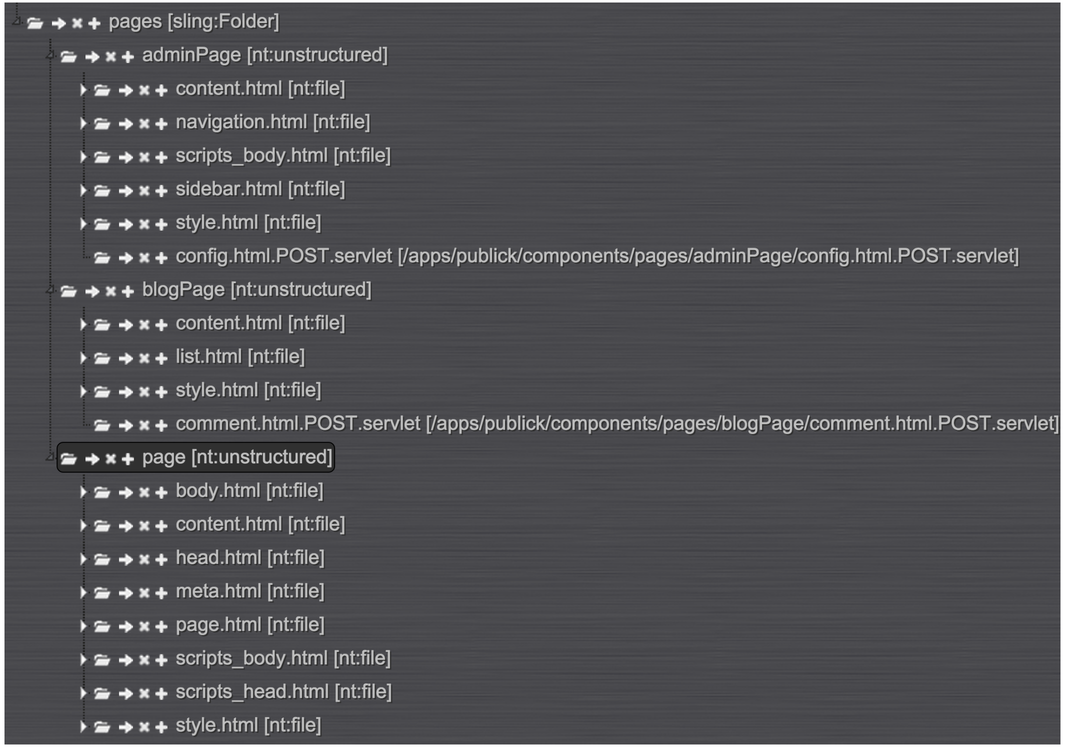
2. Avoid creating a new component from scratch if you can just extend one

3. Define some extension points, even though you might not be using them now

Modern Web Applications with Sightly

Components content structure

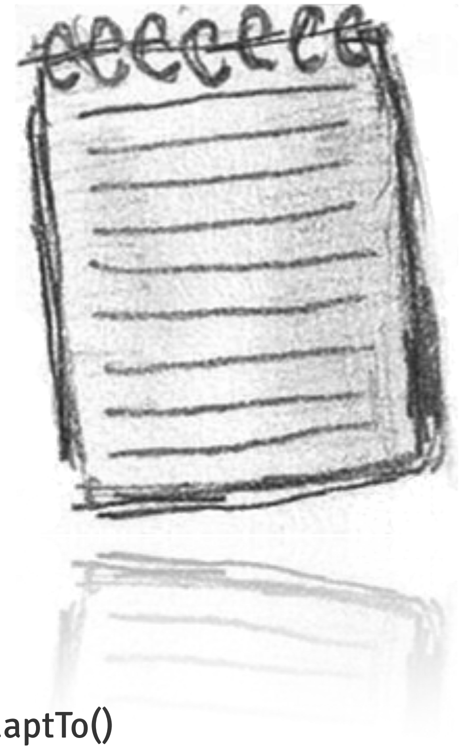




```
pages [sling:Folder]
    adminPage [nt:unstructured]
        content.html [nt:file]
        navigation.html [nt:file]
        scripts_body.html [nt:file]
        sidebar.html [nt:file]
        style.html [nt:file]
        config.html.POST.servlet [/apps/publick/components/pages/adminPage/config.html.POST.servlet]
    blogPage [nt:unstructured]
        content.html [nt:file]
        list.html [nt:file]
        style.html [nt:file]
        comment.html.POST.servlet [/apps/publick/components/pages/blogPage/comment.html.POST.servlet]
    page [nt:unstructured]
        body.html [nt:file]
        content.html [nt:file]
        head.html [nt:file]
        meta.html [nt:file]
        page.html [nt:file]
        scripts_body.html [nt:file]
        scripts_head.html [nt:file]
        style.html [nt:file]
```

Modern Web Applications with Sightly

Markup



Sightly aims to help developers write simple, uncluttered, easy to understand markup.

Ideally a template’s markup should be as close as possible to the markup that will be rendered.

Avoid thinking with a programming mindset when writing Sightly templates. Logic belongs somewhere else (hint: Use-API).

Modern Web Applications with Sightly

Markup - data-sly-text

DOs

```
<!--/* ok for placeholders */-->
<p data-sly-text="${properties.text}">Placeholder removed on
rendering.</p>

<!--/* if there are no placeholders */-->
<p>${properties.text}</p>
```

DONTs

```
<!--/* unnecessary, as you don’t gain anything */-->
<p data-sly-text="${properties.text}"></p>
```

Modern Web Applications with Sightly

Markup - data-sly-attribute

DOs

```
<!--/* use it with attribute maps */-->
<div data-sly-attribute="${component.attributes}">…</div>

<!--/* otherwise */-->
<a href="#" title="${link.title}"></a>
```

DONTs

```
<!--/* overkill */-->
<a href="#" data-sly-attribute.title="${link.title}"></a>
```

adaptTo()

Modern Web Applications with Sightly

Markup - data-sly-element

DOs

```
<!--/* context changeable elements */-->
<list data-sly-element="${list.ordered ? 'ol' : 'ul'}">…</list>

<heading data-sly-element="${component.h}">…</heading>
```

DONTs

```
<!--/* hiding logic */-->
<div data-sly-element="${comp.elem}"></div>
```

Modern Web Applications with Sightly

Markup - data-sly-test

DOs

```
<!--/* store test evaluation for if/else */-->
<div data-sly-test.author=“${wcmmode.edit}”>…</div>
<div data-sly-test=“${!author}”>…</div>
```

DONTs

```
<!--/* repeatedly call the same test expression */-->
<!--/* use it to define variables in your templates */-->
```

Modern Web Applications with Sightly

Markup - data-sly-list

DOs

```
<!--/* use it on markup that will be rendered */-->
<ul class=“fruits” data-sly-list.fruit=“${fruits}”>
    <li class=“list-item ${fruitList.odd ? ‘odd’ : ‘even’}”>
        ${fruit}
    </li>
</ul>
```

DONTs

```
<!--/* ok only if you cannot use data-sly-repeat */-->
<sly data-sly-list.paragraph=“${paragraphs}” data-sly-unwrap>
    <p>${paragraph}</p>
</sly>
```

Modern Web Applications with Sightly

Markup - data-sly-repeat (since 1.1)

DOs

```
<!--/* use it on markup that will be rendered */-->
<p data-sly-repeat.paragraph=“${paragraphs}”>${paragraph}</p>
```

Modern Web Applications with Sightly

Markup - data-sly-include

Recommendations

```
<!--/* unwrap if the tag doesn’t provide meaningful markup */-->
<!DOCTYPE html>
<html>
    <sly data-sly-include=“head.html” />
    …
</html>
```

Modern Web Applications with Sightly

Markup - data-sly-include

Recommendations

```
<!--/* however you can avoid unwrapping it if… */-->
<!DOCTYPE html>
<html>
    <head data-sly-include=“head.html”></head>
    …
</html>

head.html:
<title>${properties.title || properties[‘jcr:title’]}</title>
```

Modern Web Applications with Sightly

Markup - `data-sly-resource`

Recommendations

```
<!--/* same recommendations as for data-sly-include */-->
```

Modern Web Applications with Sightly

Markup - data-sly-use

DOs

```
<!--/* integrate it on the top-most tag where you need it; avoid
unwrapping its container tag */-->
<ul data-sly-use.var="${..}" data-sly-list="${var.list}">…
```

DONTs

```
<!--/* use it in loops */-->
<!--/* declare all your objects at the top of the script, on <sly>
tags or using data-sly-unwrap */-->
```

Modern Web Applications with Sightly

Markup - `data-sly-unwrap / <sly>`

DOs

```
<!--/* use it sparingly, only when there’s no other option */-->
<head data-sly-use.clientlib="/libs/granite/sightly/templates/
clientlib.html">
    <sly data-sly-call="${clientLib.css @
categories='foundation'}" />
```

DONTs

```
<!--/* use it to remove markup that shouldn’t have been there in
the first place */-->
```

Modern Web Applications with Sightly

Markup - the context option

DOs

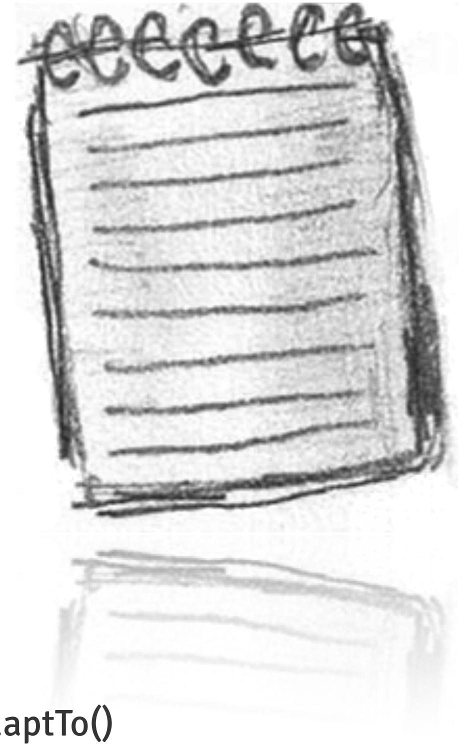
```
<!--/* use it carefully, when you really know what you're
doing*/-->
<div data-type="comment" data-path="${comment.path @
context='uri'}">…
```

DONTs

```
<!--/* use context='unsafe' if actually a better value could be
used */-->
```

Modern Web Applications with Sightly

Use-API

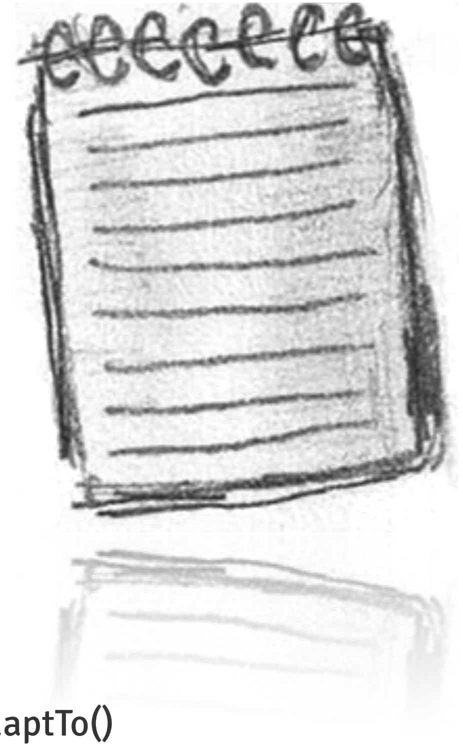


It’s the only way to load helper objects for your Sightly scripts.

While the specification only mentions templates, Java and JavaScript objects, the API’s implementation from Sling is much more powerful.

Modern Web Applications with Sightly

Use-API

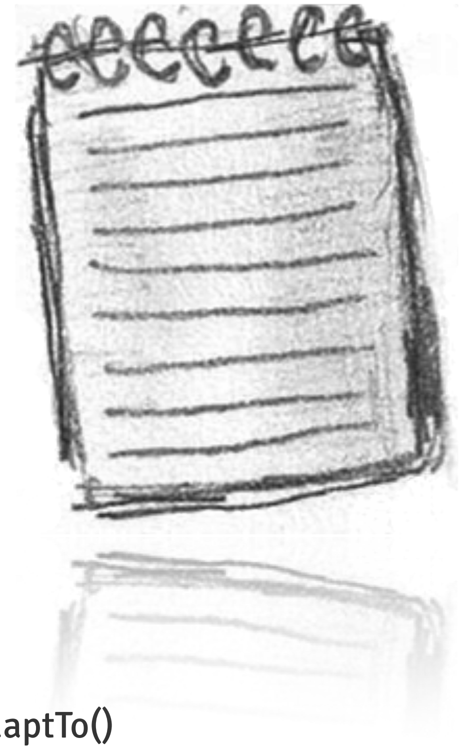


In Sling the Use-API can load:

1. Sling Models (they're really cool to use!)
2. Java objects (whether they are OSGi services, are adaptable from `SlingHttpServletRequest`, `Resource`, implement `Use` or not, exposed from bundles or stored in the repository)
3. JavaScript objects, through the `use` function
4. Any other script evaluated by a Script Engine from Sling

Modern Web Applications with Sightly

Use-API - what’s the best option?

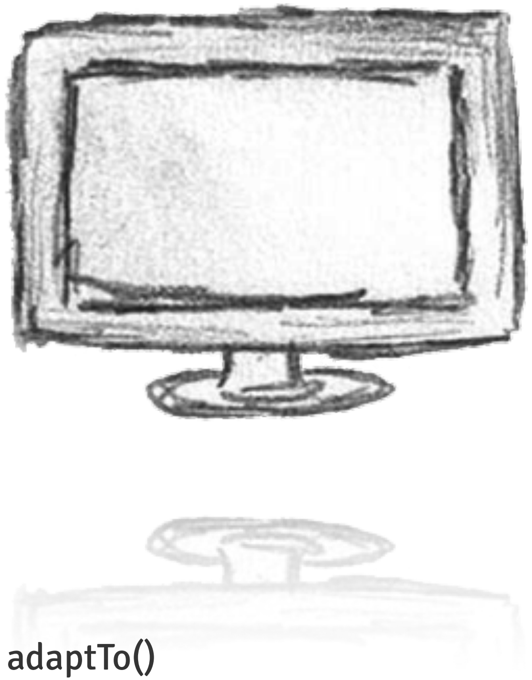


If the logic is not strictly tied to a component and the Use-object is reusable between scripts:
➡ Java object stored in an OSGi bundle or a Sling Model (dependency injection FTW)

If the logic is specific to a component:
➡ Java POJO stored in the repository, for best performance
➡ JavaScript stored in the repository, if your components are maintained mostly by front-end developers

Use objects in the repository should still be treated as API!

Best practices on slides. Do we have an app?



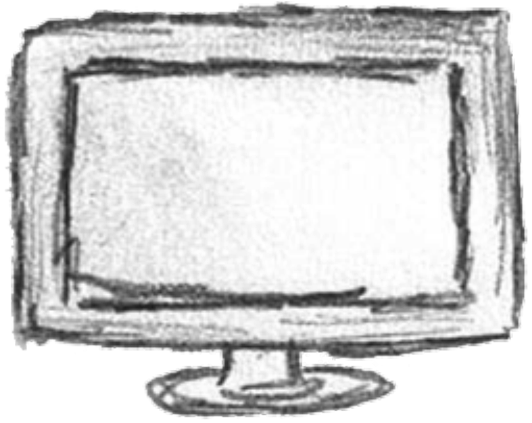
Yes. It’s called Publick.

Initially developed by Nate Yolles, to host host his blog:
https://github.com/nateyolles/publick-sling-blog.git

Forked to implement best practices at:
https://github.com/raducotescu/publick-sling-blog.git

Purpose:
Commit it to Sling, as a best practices Sightly + other Sling goodies fully functional demo application.

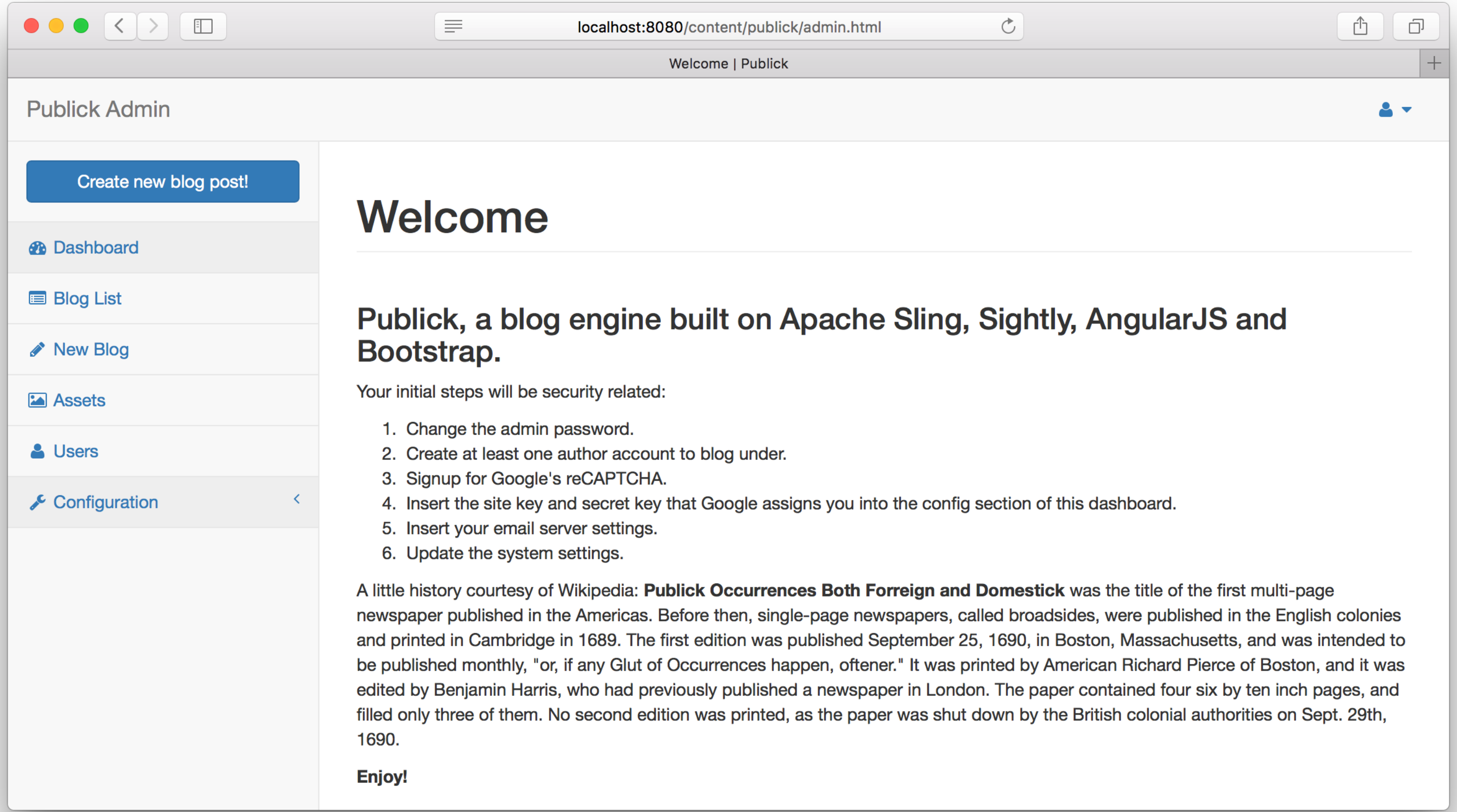
What is Publick?



Yet another blog engine built on top of Apache Sling, Sightly, AngularJS and Bootstrap.

AngularJS is only used for creating an interaction with existing Sling services (mostly the user admin from `/system/userManager`).

Modern Web Applications with Sightly



Welcome

Publick, a blog engine built on Apache Sling, Sightly, AngularJS and Bootstrap.

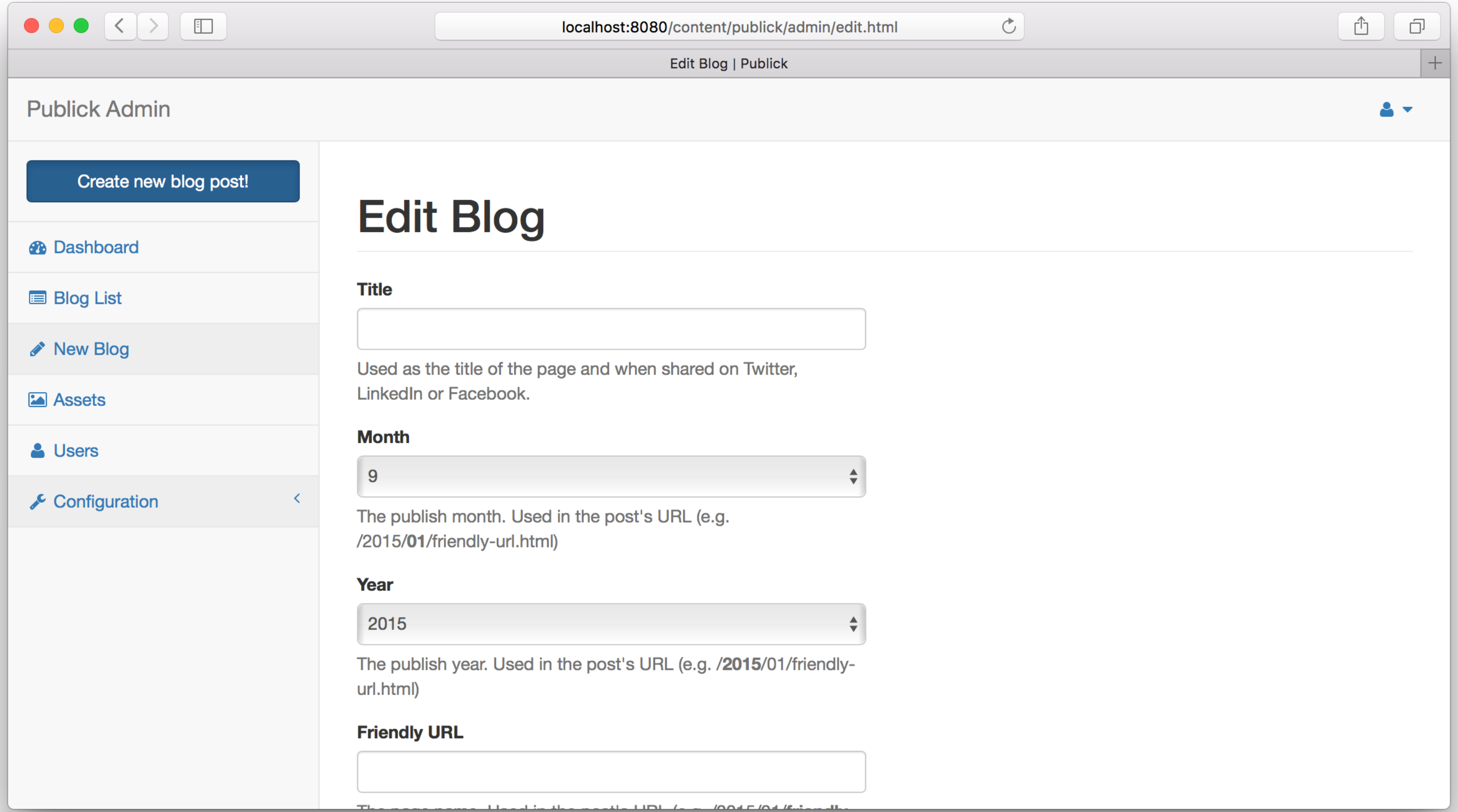
Your initial steps will be security related:

1. Change the admin password.
2. Create at least one author account to blog under.
3. Signup for Google's reCAPTCHA.
4. Insert the site key and secret key that Google assigns you into the config section of this dashboard.
5. Insert your email server settings.
6. Update the system settings.

A little history courtesy of Wikipedia: **Publick Occurrences Both Forreign and Domestick** was the title of the first multi-page newspaper published in the Americas. Before then, single-page newspapers, called broadsides, were published in the English colonies and printed in Cambridge in 1689. The first edition was published September 25, 1690, in Boston, Massachusetts, and was intended to be published monthly, "or, if any Glut of Occurrences happen, oftener." It was printed by American Richard Pierce of Boston, and it was edited by Benjamin Harris, who had previously published a newspaper in London. The paper contained four six by ten inch pages, and filled only three of them. No second edition was printed, as the paper was shut down by the British colonial authorities on Sept. 29th, 1690.

Enjoy!

Modern Web Applications with Sightly



localhost:8080/content/publick/admin/edit.html

Edit Blog | Publick

Publick Admin

Create new blog post!

- Dashboard
- Blog List
- New Blog
- Assets
- Users
- Configuration

Edit Blog

Title

Used as the title of the page and when shared on Twitter, LinkedIn or Facebook.

Month

9

The publish month. Used in the post's URL (e.g. /2015/**01**/friendly-url.html)

Year

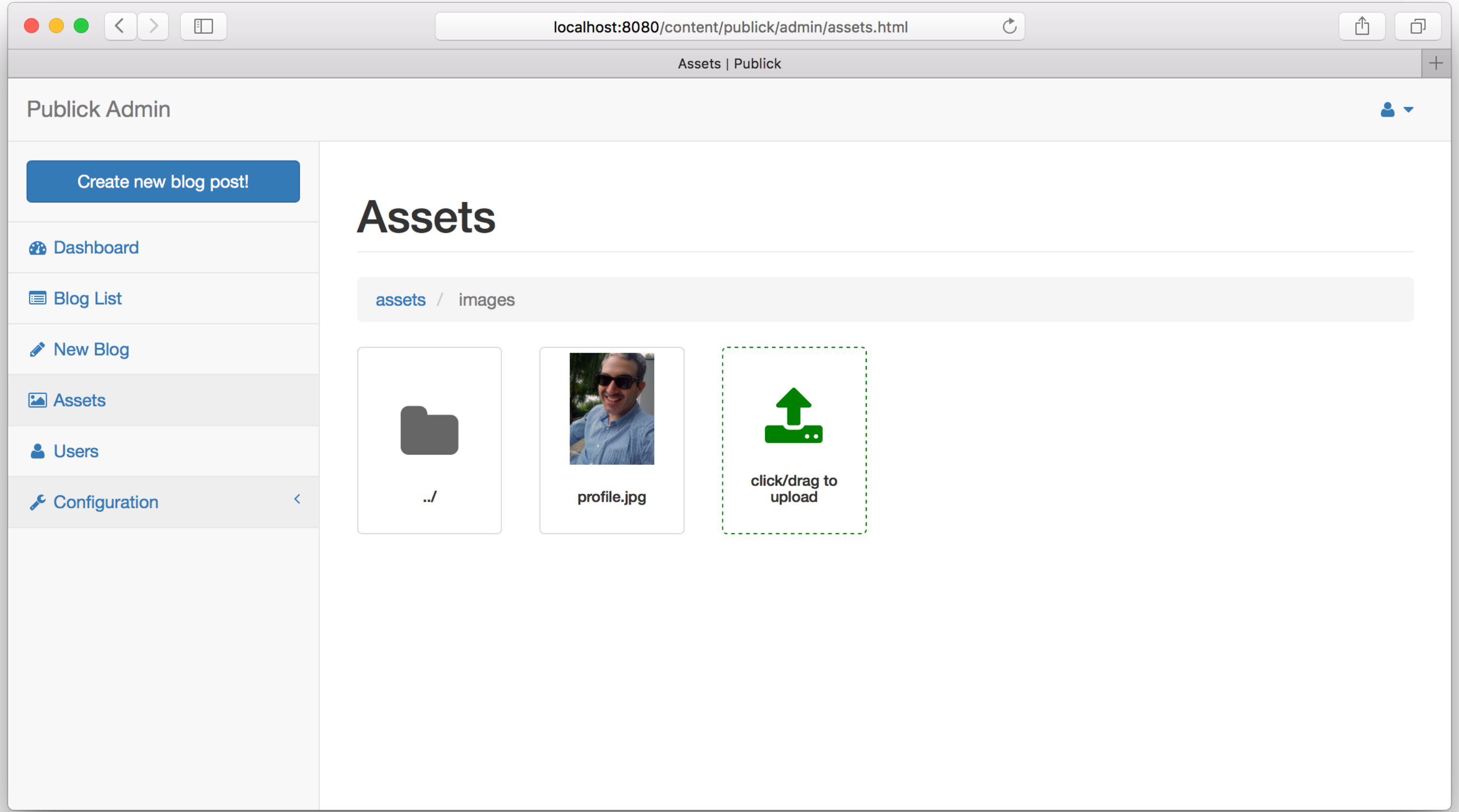
2015

The publish year. Used in the post's URL (e.g. /**2015**/01/friendly-url.html)

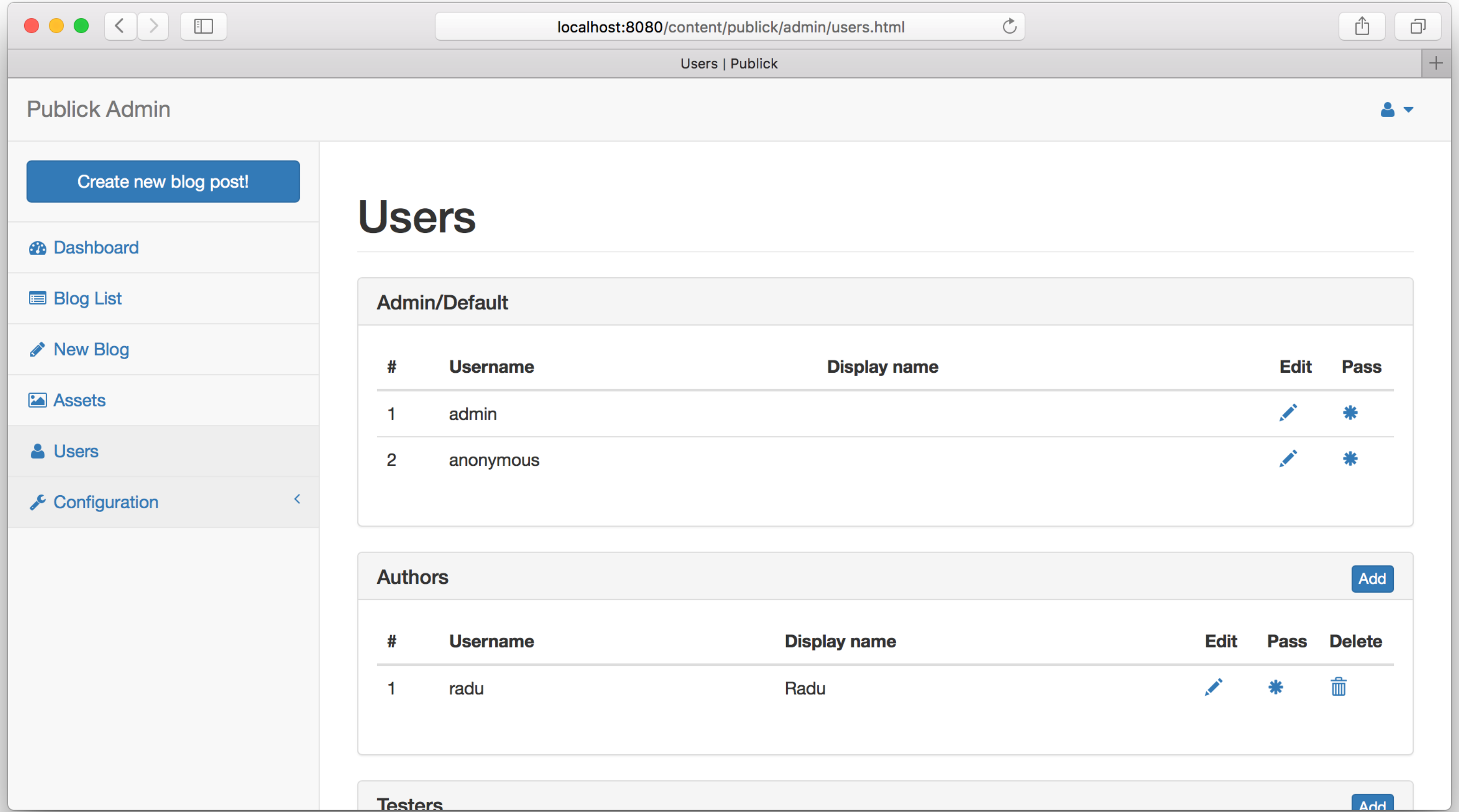
Friendly URL

adaptTo()

Modern Web Applications with Sightly



Modern Web Applications with Sightly



localhost:8080/content/publick/admin/users.html

Users | Publick

Publick Admin

Create new blog post!

- Dashboard
- Blog List
- New Blog
- Assets
- Users
- Configuration

Users

Admin/Default

#	Username	Display name	Edit	Pass
1	admin		✎	✱
2	anonymous		✎	✱

Authors

Add

#	Username	Display name	Edit	Pass	Delete
1	radu	Radu	✎	✱	🗑

Testers

Add

Modern Web Applications with Sightly

Questions?

Modern Web Applications with Sightly

Credits & resources

1. https://github.com/Adobe-Marketing-Cloud/sightly-spec/blob/master/SPECIFICATION.md - Sightly HTML templating language specification
2. https://github.com/nateyolles/publick-sling-blog - original version of Publick
3. Hand-drawn icons from https://www.iconfinder.com/iconsets/49handdrawing