Development Best Practices

AEM Developer Meetup
Utrecht, September 3rd 2015

@GabrielWalt, Product Manager

Best Practices…

Best practices are useful reference points, but they must come with a warning label : The more you rely on external intelligence, the less you will value an internal idea.

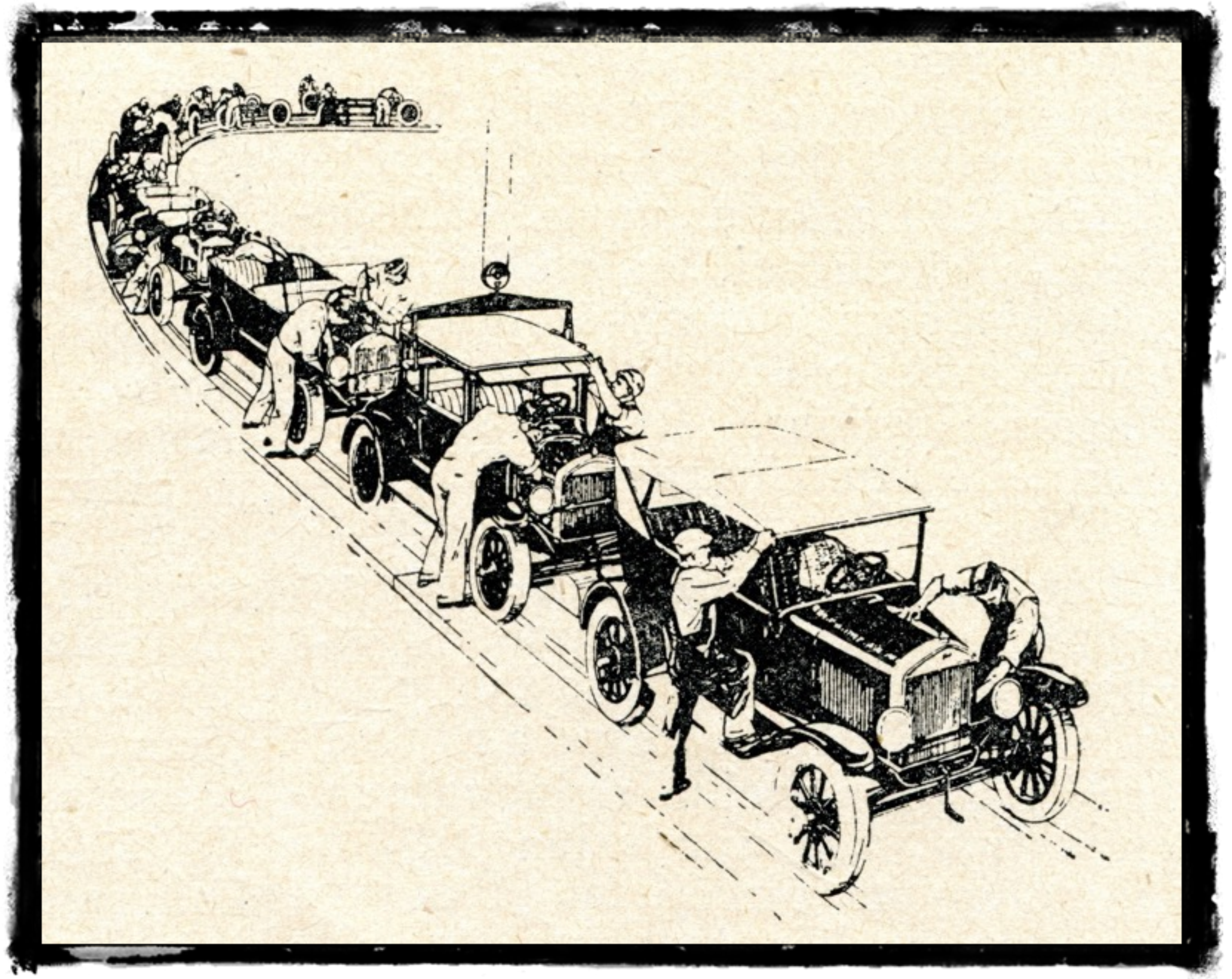
And this is the age of the idea.

— Gyan Nagpal

Best Practices...

Practices that lead to more efficiency

- **Less project effort**
- **Less operational effort**
- **Less maintenance effort**

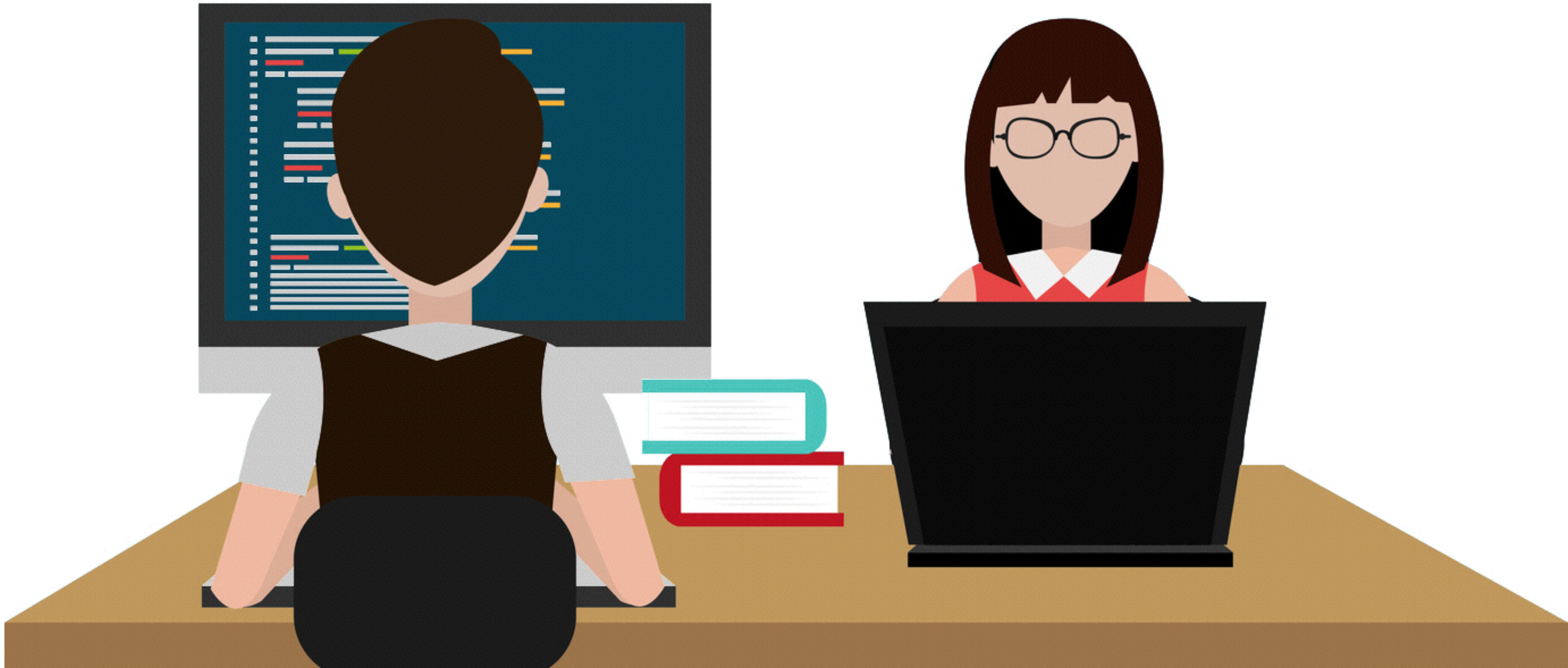


Automotive assembly line, ca. 1920

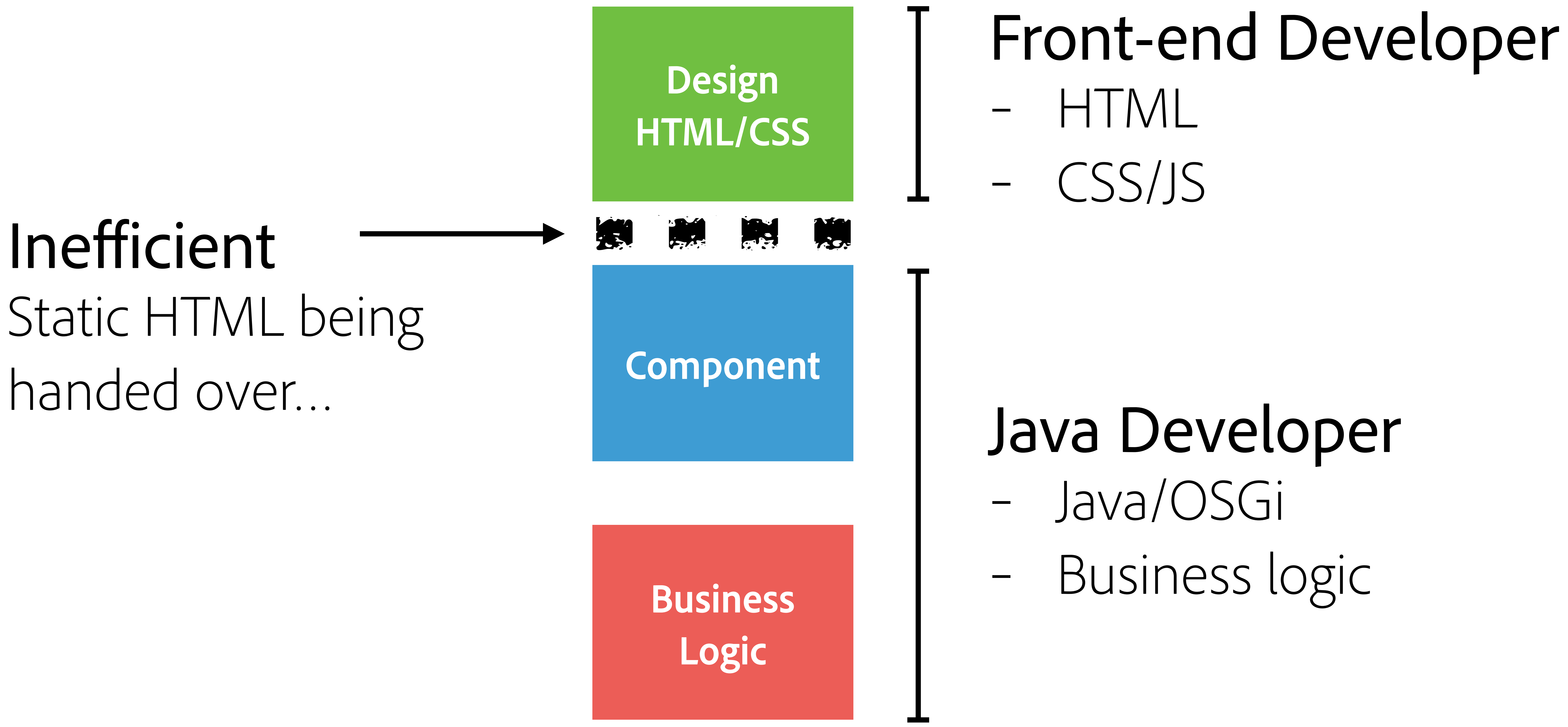
Separation of Concerns

Front-end Developer
- HTML
- CSS/JS

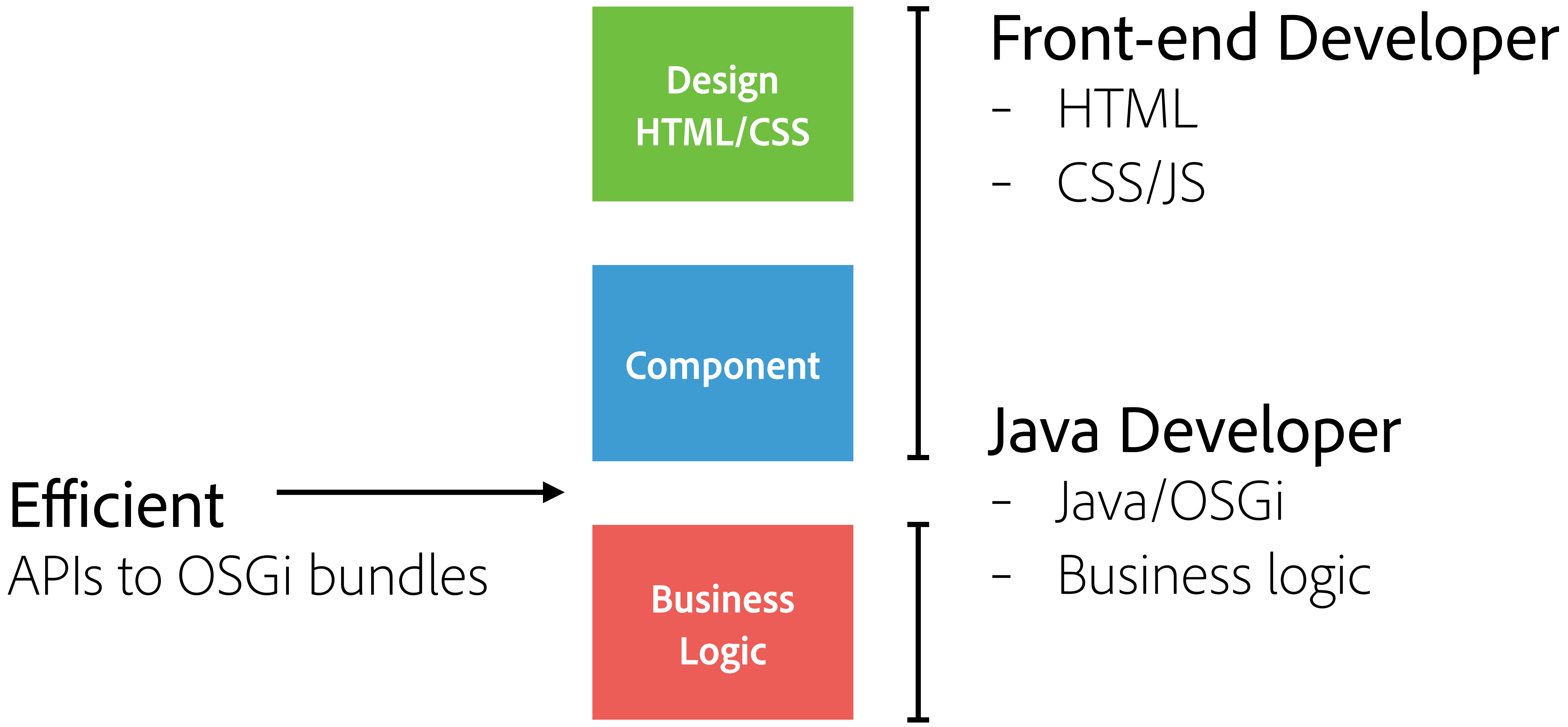
Java Developer
- Java/OSGi
- Business logic



Separation of Concerns



Separation of Concerns



Front-end Developer
- HTML
- CSS/JS

Java Developer
- Java/OSGi
- Business logic

Efficient
APIs to OSGi bundles

§1 – Separating concerns

http://docs.adobe.com/content/docs/en/aem/6-1/develop/sightly.html

sightly

BEAUTIFUL MARKUP

Sightly Template Language

```
<a href="${properties.link}" data-sly-test="${properties.jcr:title}">
    ${properties.jcr:title}
</a>
```

- Code-less language, forcing strict separation of concerns
- Powerful extension points with the Use-API
- Automatic contextual HTML escaping and XSS protection
- Automatically removes HTML attributes if value is empty
- Reuses HTML blocks for statements
- On-par performance and scalability with JSP

Use Statement

Initializes a helper object.

```
<div data-sly-use.logic="logic.js">${logic.hi}</div>
```

Output:

```
<div>Hello World</div>
```

Server-side JavaScript logic

```
<!-- template.html -->
<div data-sly-use.logic="logic.js">${logic.hi}</div>


/* logic.js */
use(function () {
   return {
       hi: "Hello World"
    };
});
```

Java logic

Adaptable with SlingModels

```
<!-- template.html -->
<div data-sly-use.logic="com.myorg.foo.Logic">${logic.hi}</div>

/* Logic.java in OSGi bundle */
package com.myorg.foo;
import javax.annotation.PostConstruct;
import org.apache.sling.api.resource.Resource;
import org.apache.sling.models.annotations.Model;

@Model(adaptables = Resource.class)
public class Logic {
    private String hi;

    @PostConstruct
    protected void init() {
        hi = "Hello World";
    }

    public String getHi() {
        return hi;
    }
}
```

The Use-API accepts classes that are adaptable from Resource or Request.

What kind of Use-API?

Model logic

This logic is not tied to a template and is potentially reusable among components.
It should aim to form a stable API that changes little, even in case of a full redesign.

➔ Java located in OSGi bundle

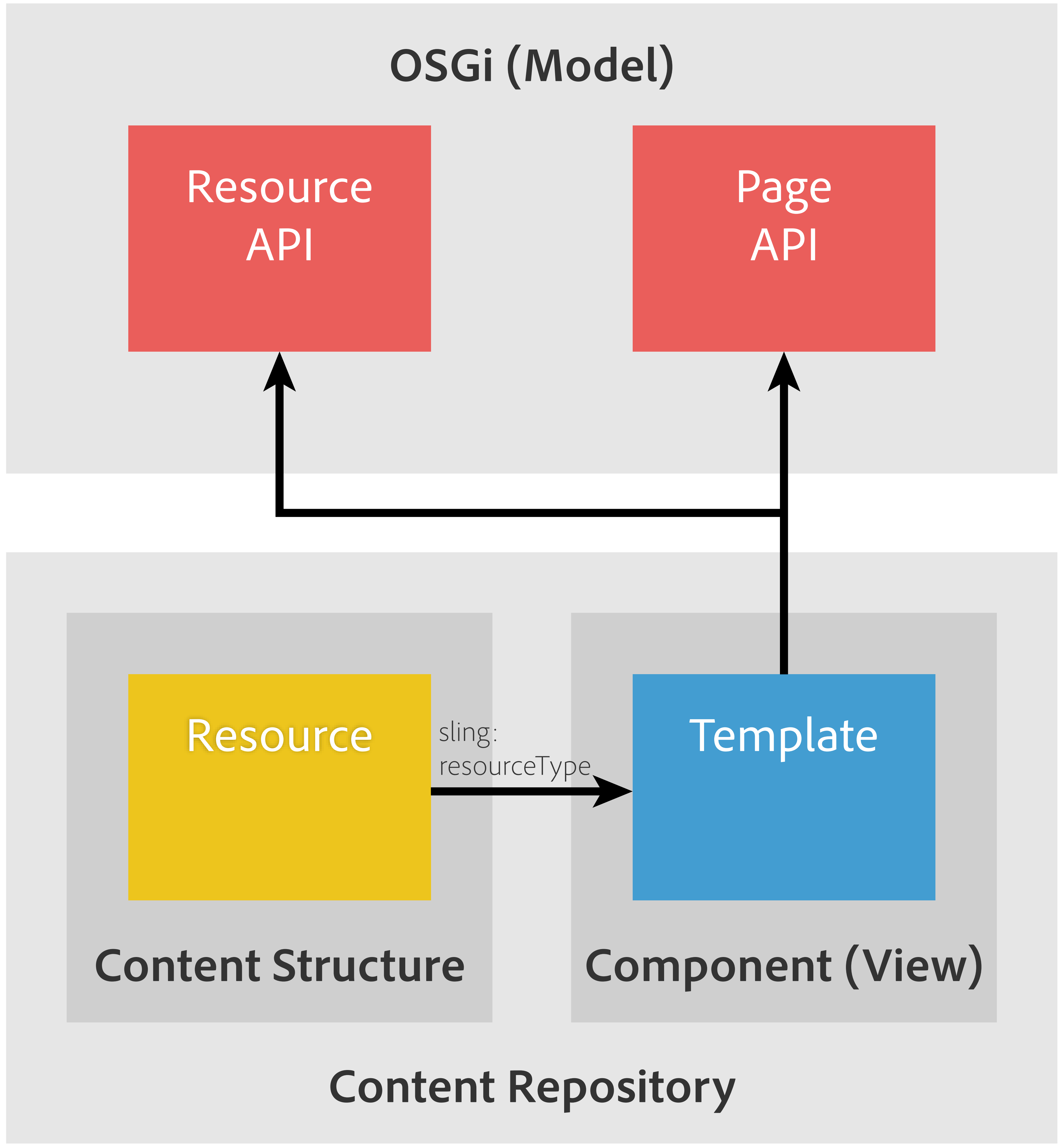
View logic

This logic is specific to the templates and is likely to change if the design changes.
It is thus a good practice to locate it in the content repository, next to the template.

➔ JavaScript located in component
If components are to be maintained by front-end devs (typically with Brackets).

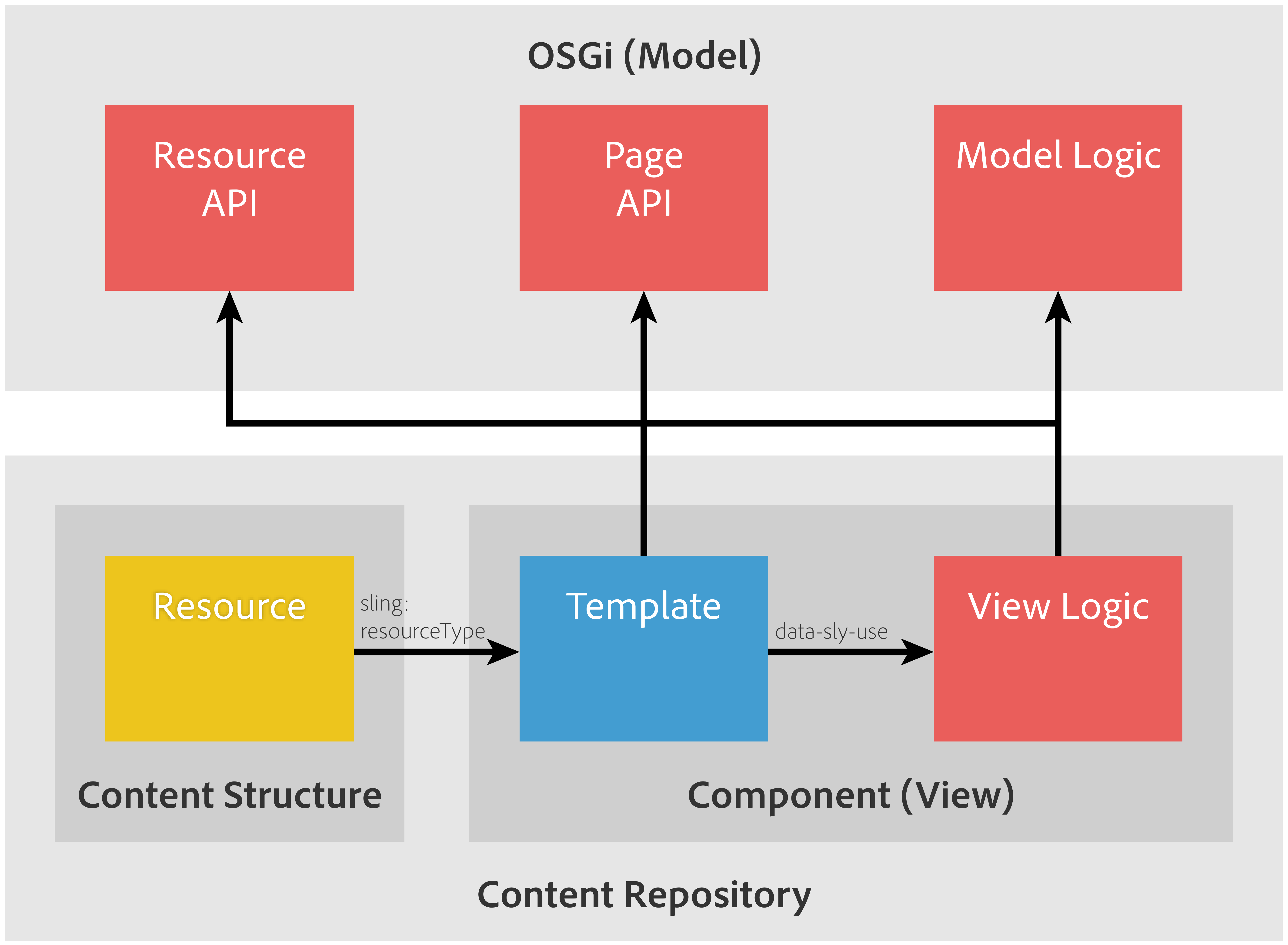
➔ Java located in component
If performance is critical (e.g. when many requests are not cached by the dispatcher).

Start simple: first, no code!



- Sling plays the role of the **controller** and resolves the sling:resourceType, deciding which component will render the accessed resource.
- The component plays the role of the **view** and it's Sightly template builds the corresponding markup.
- The Resource and Page APIs play the role of the **model**, which are available from the template as variables.

Add logic only where needed

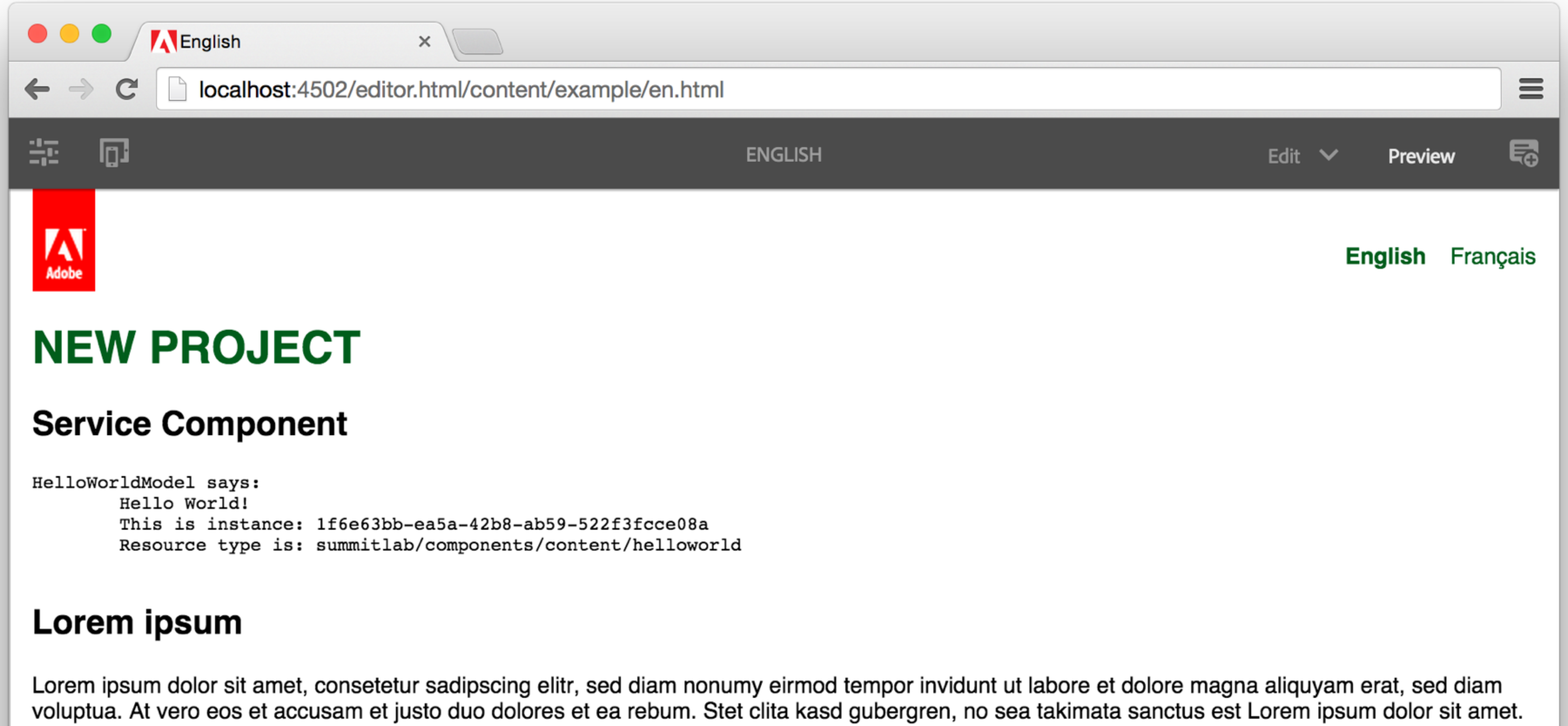


- **Model Logic** is needed only if the logic to access the data is different to what existing APIs provide.
- **View Logic** is needed only when the template needs additional data preparation.

§2 – Enabling the Java Developer

- Getting started
  The AEM Project Archetype
- Working efficiently
  The AEM Developer Tools

AEM Project Archetype



NEW PROJECT

Service Component

```
HelloWorldModel says:
        Hello World!
        This is instance: 1f6e63bb-ea5a-42b8-ab59-522f3fcce08a
        Resource type is: summitlab/components/content/helloworld
```

Lorem ipsum

Lorem ipsum dolor sit amet, consetetur sadipscing elitr, sed diam nonumy eirmod tempor invidunt ut labore et dolore magna aliquyam erat, sed diam voluptua. At vero eos et accusam et justo duo dolores et ea rebum. Stet clita kasd gubergren, no sea takimata sanctus est Lorem ipsum dolor sit amet.

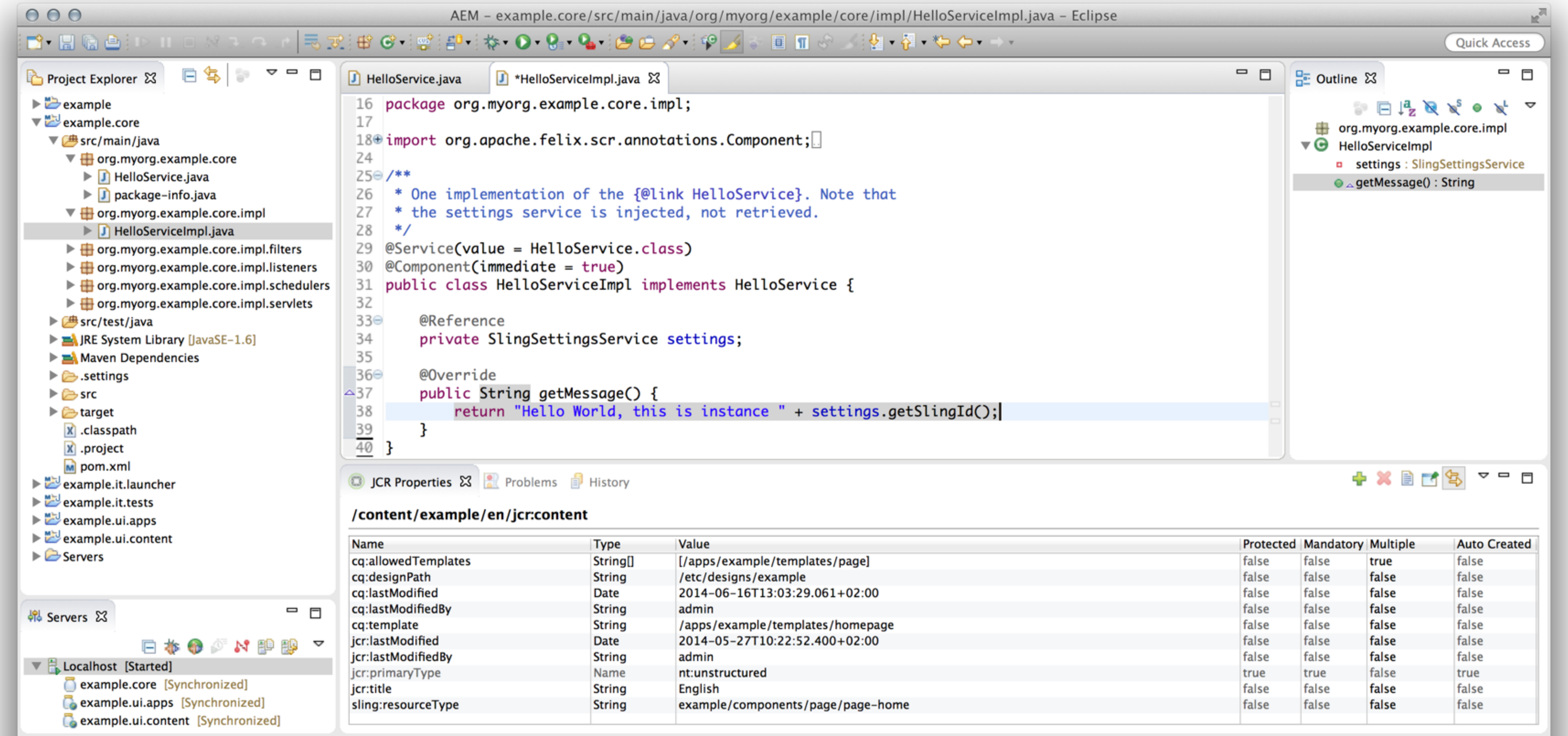
AEM Project Archetype

https://github.com/Adobe-Marketing-Cloud/aem-project-archetype

Creates a new project with latest best practices prepared

- Separate project structure for bundles, apps, content and tests.
- Sightly components super-typed in apps with corresponding client-libraries.
- OSGi config folder, asset d&d, device emulator, dictionary structure.
- Bundle examples for Sling Models, Servlets, Filters and Schedulers.
- Unit tests, integration tests, and client-side Hobbes tests with dev mode.

AEM Developer Tools



AEM – example.core/src/main/java/org/myorg/example/core/impl/HelloServiceImpl.java – Eclipse

```
▶ example
▼ example.core
  ▼ src/main/java
    ▼ org.myorg.example.core
      ▶ HelloService.java
      ▶ package-info.java
    ▼ org.myorg.example.core.impl
      ▶ HelloServiceImpl.java
    ▶ org.myorg.example.core.impl.filters
    ▶ org.myorg.example.core.impl.listeners
    ▶ org.myorg.example.core.impl.schedulers
    ▶ org.myorg.example.core.impl.servlets
  ▶ src/test/java
  ▶ JRE System Library [JavaSE-1.6]
  ▶ Maven Dependencies
  ▶ .settings
  ▶ src
  ▶ target
    .classpath
    .project
    pom.xml
▶ example.it.launcher
▶ example.it.tests
▶ example.ui.apps
▶ example.ui.content
▶ Servers
```

```java
package org.myorg.example.core.impl;

import org.apache.felix.scr.annotations.Component;

/**
 * One implementation of the {@link HelloService}. Note that
 * the settings service is injected, not retrieved.
 */
@Service(value = HelloService.class)
@Component(immediate = true)
public class HelloServiceImpl implements HelloService {

    @Reference
    private SlingSettingsService settings;

    @Override
    public String getMessage() {
        return "Hello World, this is instance " + settings.getSlingId();
    }
}
```

```
org.myorg.example.core.impl
▼ HelloServiceImpl
    settings : SlingSettingsService
    getMessage() : String
```

```
▼ Localhost  [Started]
    example.core  [Synchronized]
    example.ui.apps  [Synchronized]
    example.ui.content  [Synchronized]
```

/content/example/en/jcr:content

Name	Type	Value	Protected	Mandatory	Multiple	Auto Created
cq:allowedTemplates	String[]	[/apps/example/templates/page]	false	false	true	false
cq:designPath	String	/etc/designs/example	false	false	false	false
cq:lastModified	Date	2014-06-16T13:03:29.061+02:00	false	false	false	false
cq:lastModifiedBy	String	admin	false	false	false	false
cq:template	String	/apps/example/templates/homepage	false	false	false	false
jcr:lastModified	Date	2014-05-27T10:22:52.400+02:00	false	false	false	false
jcr:lastModifiedBy	String	admin	false	false	false	false
jcr:primaryType	Name	nt:unstructured	true	true	false	true
jcr:title	String	English	false	false	false	false
sling:resourceType	String	example/components/page/page-home	false	false	false	false

AEM Developer Tools

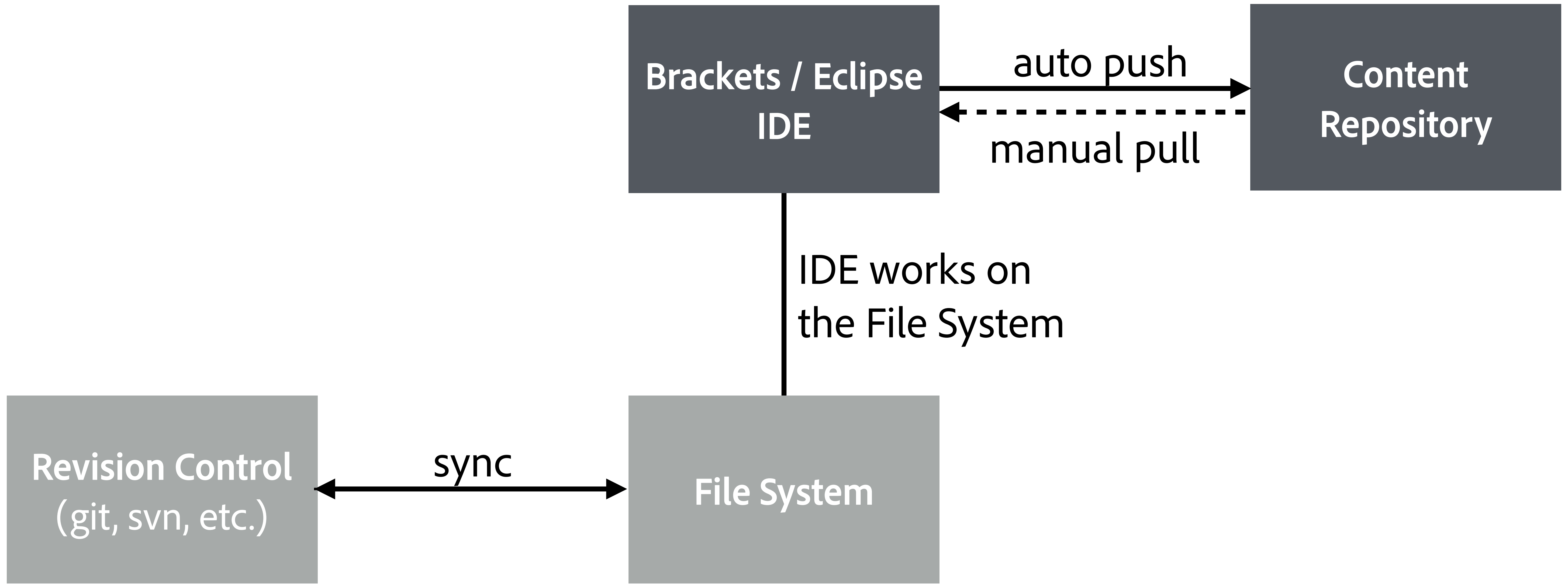
https://docs.adobe.com/docs/en/dev-tools/aem-eclipse.html

Gets new Java developers quickly efficient with AEM

- Simple bootstrap of AEM projects via a specific Project Creation Wizard.
- Easy synchronization for both content and OSGI bundles.
- Seamless integration with AEM instances through Eclipse Server Connector.
- Debugging support with code hot-swaping capabiliby.
- JCR properties edition.

AEM Developer Tools Sync

Work on file system + transparent sync & content editing

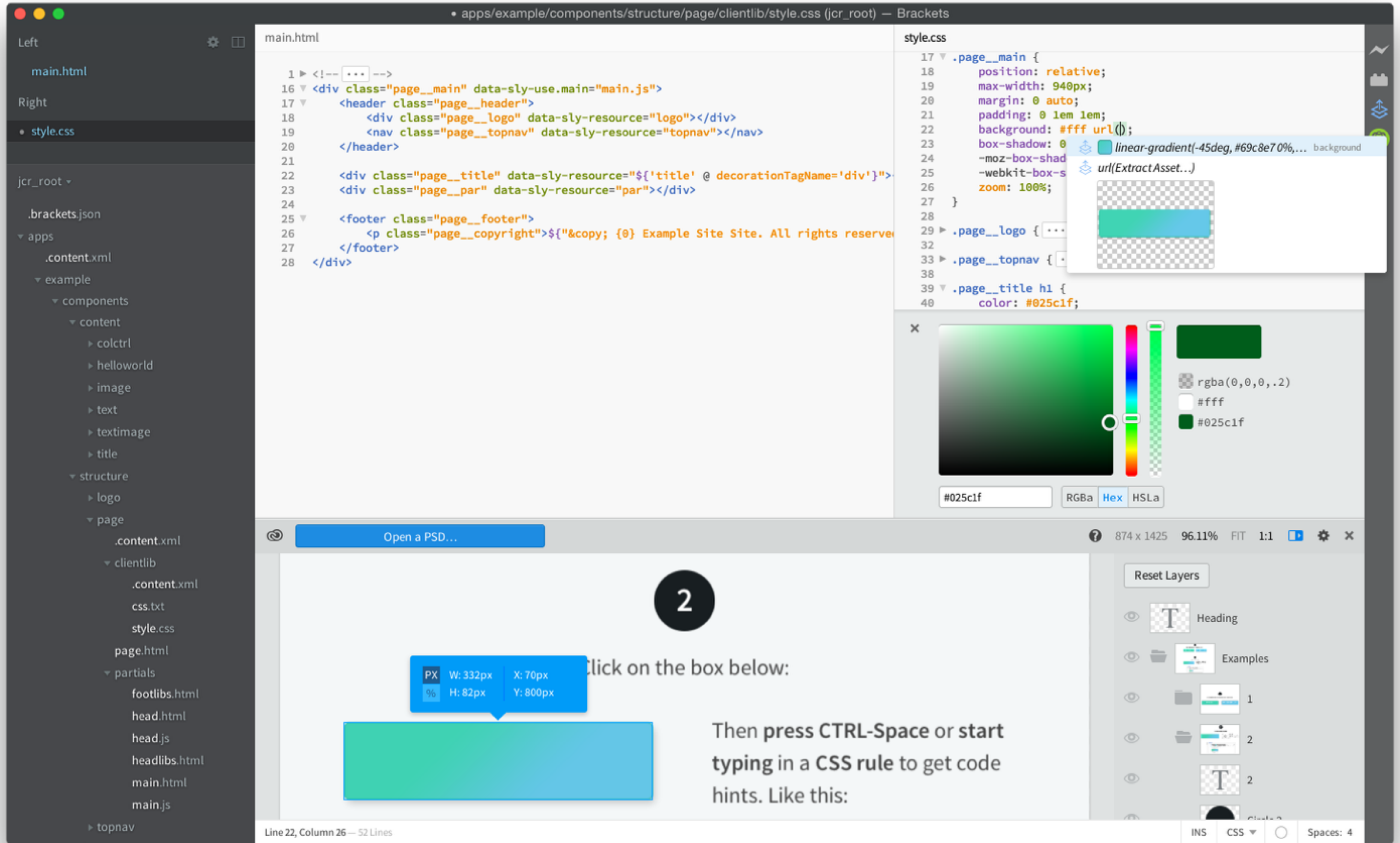


§3 – Enabling the Front-End Dev

- Efficiently converting designs to web
  Brackets and the Extract extension

- Working efficiently on AEM projects
  Brackets and the AEM extension

Brackets



main.html

```
<!-- ••• -->
<div class="page__main" data-sly-use.main="main.js">
    <header class="page__header">
        <div class="page__logo" data-sly-resource="logo"></div>
        <nav class="page__topnav" data-sly-resource="topnav"></nav>
    </header>

    <div class="page__title" data-sly-resource="${'title' @ decorationTagName='div'}">
    <div class="page__par" data-sly-resource="par"></div>

    <footer class="page__footer">
        <p class="page__copyright">${"&copy; {0} Example Site Site. All rights reserve
    </footer>
</div>
```

style.css

```
.page__main {
    position: relative;
    max-width: 940px;
    margin: 0 auto;
    padding: 0 1em 1em;
    background: #fff url();
    box-shadow: 0
    -moz-box-shad
    -webkit-box-s
    zoom: 100%;
}

.page__logo { •••

.page__topnav { •

.page__title h1 {
    color: #025c1f;
```

```
linear-gradient(-45deg, #69c8e7 0%,...   background
url(ExtractAsset...)
```

```
rgba(0,0,0,.2)
#fff
#025c1f
#025c1f
```

Click on the box below:

Then **press CTRL-Space** or **start typing** in a **CSS rule** to get code hints. Like this:

Brackets

https://docs.adobe.com/docs/en/dev-tools/aem-brackets.html

Brackets and the Extract extension

- Photoshop file support to extract information from a PSD file.
- Code hints from the PSD, to easily reuse this extracted information in the code.
- CSS preprocessor support, like LESS and SCSS.
- And hundreds of additional extensions that cover more specific needs.

Brackets

https://docs.adobe.com/docs/en/dev-tools/aem-brackets.html

Brackets and the AEM extension

- Automated synchronization of changed files to the AEM development instance.
- Manual bidirectional synchronization of files and folders.
- Full content-package synchronization of the project.
- Sightly code completion for expressions and `data-sly-*` block statements.

Additional words of wisdom

- Milage may vary, cultivate critical thinking.
- Don’t mix concerns, stick to the language of the file extension.
- Resist complexity, over-architecting is just moving the problem.
- Keep it simple, it’s just a web server.

Thank you!

Developer tools:

- Project Archetype
  https://github.com/Adobe-Marketing-Cloud/aem-project-archetype
- AEM Eclipse Extension
  https://docs.adobe.com/docs/en/dev-tools/aem-eclipse.html
- AEM Brackets Extension
  https://docs.adobe.com/docs/en/dev-tools/aem-brackets.html
- Sightly Template Language
  http://www.slideshare.net/GabrielWalt/component-development
- Sightly REPL Tool
  https://github.com/Adobe-Marketing-Cloud/aem-sightly-repl
- Sightly TodoMVC Example
  https://github.com/Adobe-Marketing-Cloud/aem-sightly-sample-todomvc

Adobe®