



# Test-Driven Approaches to Documentation with Gradle, Asciidoctor, and Exemplar

# Follow Along



[jlstrater.github.io/testdriven-docs-microxhg](https://jlstrater.github.io/testdriven-docs-microxhg) | pdf

# Outline

- Background
- Documentation in General
- Intro to Asciidoc (and Asciidoctor)
- Test-Driven Approaches and Exemplar

# Who am I?

```
speaker {  
    name 'Jennifer "Jenn" Strater'  
    company 'Gradle Inc'  
    ossContributions 'Codenarc, Spring REST Docs, and more',  
    communityWork 'Groovy Community Slack owner',  
        'GR8DI board',  
        'GR8Conf crew',  
    twitter '@codeJENNerator',  
    github 'jlstrater',  
    extraDescription '''always looking for more time  
    to contribute to Open Source'''  
}
```

# Before Gradle

- Groovy (and Grails) dev in the US and Germany
- Experience in Microservice and Monolithic Architectures
- Spring REST Docs contributor and advocate

# What is Gradle?

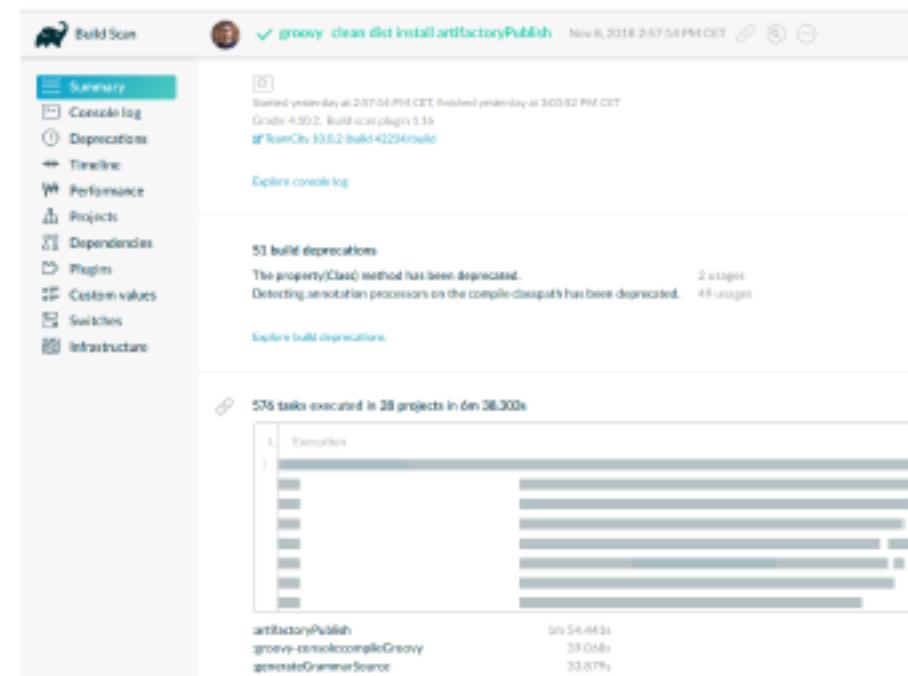


Gradle is a build and automation tool.

- JVM based
- Implemented in Java
- 100% Free Open Source - Apache Standard License 2.0
- Supports Java (inc. Groovy, Kotlin, Scala), Native (C, C++, Swift), Android, Go, Asciidoctor, and more

# Gradle Inc.

- Build Happiness
- Employs full time engineers
- Provide Gradle build scans and build caching in Gradle Enterprise for both Gradle AND Maven.
- Try it for free at [scans.gradle.com](https://scans.gradle.com)





- On premise build cache and build scan solution
- Out of the box solution for distributed build cache
- Improved build scan feature set
  - Searchable build history
  - Build comparison

# Gradle is hiring!

- Fully distributed development team
- Exciting project used by millions
- Build tool team and Gradle enterprise positions

If anything you hear from now on sounds like a great problem to solve,

[gradle.com/careers](https://gradle.com/careers)

# Talk to me!

- About Gradle
- About Gradle Enterprise
- About Careers at Gradle GmbH
- If you just want a Gradlephant sticker



# About you

# About you

- Ecosystem (Java, Android, JS, etc)

# About you

- Ecosystem (Java, Android, JS, etc)
- Monolith, Microservice architecture

# About you

- Ecosystem (Java, Android, JS, etc)
- Monolith, Microservice architecture
- Role in Documentation (writer, maintainer)



microchg<sup>2019</sup>



I hate writing documentation!\*

# Documentation

# Type of information being conveyed

- Is this a reference document?
- Does it belongs in the user manual?
- Can it be a separate, independent document like a tutorial?

# Who owns the process?

# Who owns the process?

- Developers

# Who owns the process?

- Developers
- Product teams

# Who owns the process?

- Developers
- Product teams
- Writers

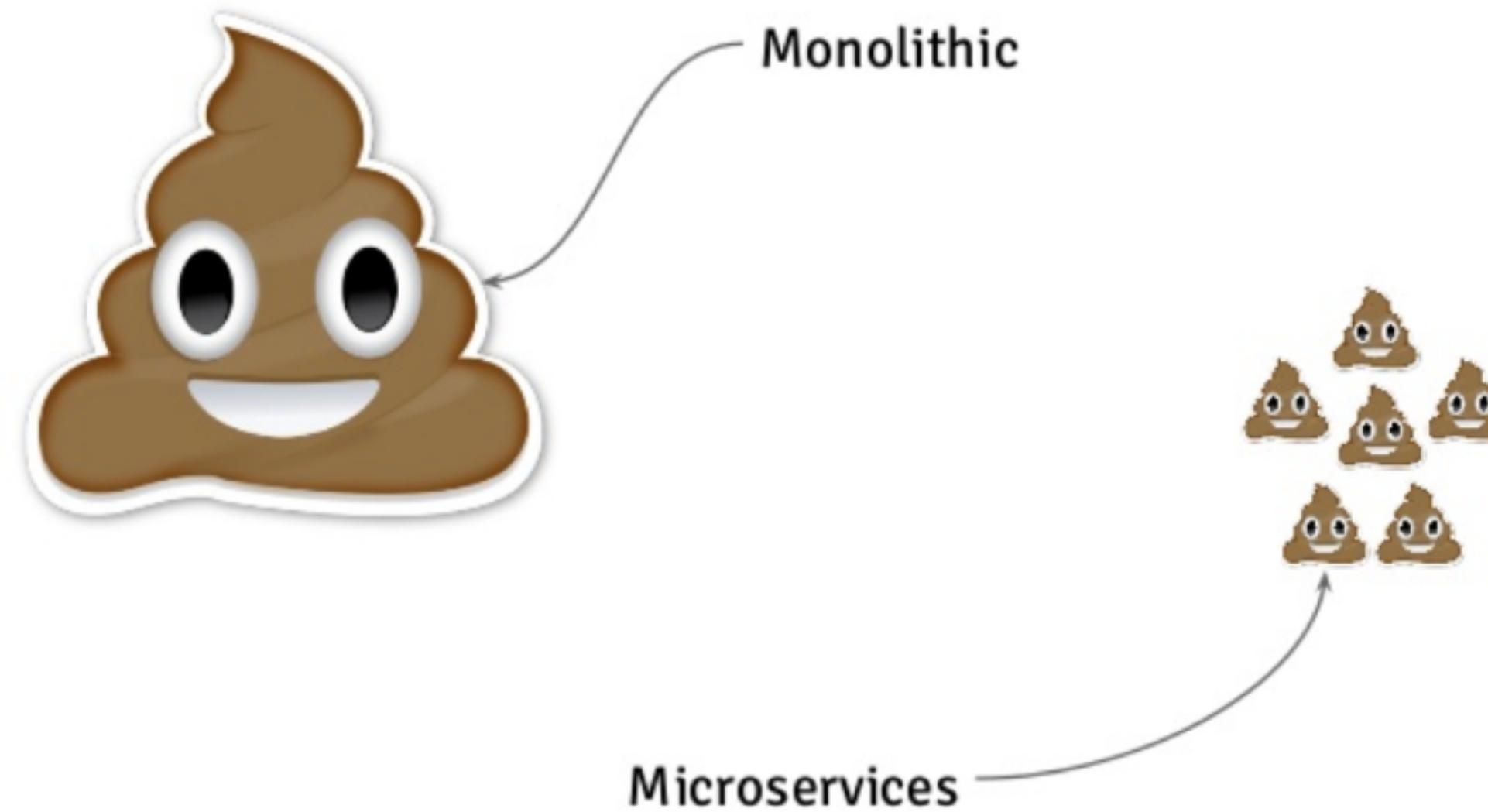
# Versioning

- Same as product or separate/"current"
- They need to be versioned just like the source code so they refer to historical features/implementations.
- Docs cannot be updated dynamically

Out of date docs are worse than no  
docs at all.

# Solve your problems now!

## Monolithic vs Microservices



Attribution: @Alvaro\_Sanchez

# Types of documentation at Gradle

# Reference

The screenshot shows the top navigation bar of the Gradle Docs 5.3.1 website. It includes links for Community, Training, News, Enterprise, and a search icon. Below the navigation is a menu bar with links for OVERVIEW, PACKAGE, CLASS, TREE, DEPRECATED, INDEX, and HELP. A search bar labeled "SEARCH:" with a magnifying glass icon is also present.

## Gradle API 5.3.1

Packages	Description
org.gradle	Classes for embedding Gradle.
org.gradle.api	Start Here: Gradle's <b>Project API</b> , which is available from your build files.
org.gradle.api.artifacts	Classes for declaring and using artifacts and artifact dependencies.
org.gradle.api.artifacts.component	Classes that provide meta-data about software components.
org.gradle.api.artifacts.dsl	Classes used in the artifact DSL.
org.gradle.api.artifacts.ivy	Classes for declaring and using Ivy modules.
org.gradle.api.artifacts.maven	Maven specific classes for dependency management.
org.gradle.api.artifacts.query	Classes used for querying the artifacts.

- Javadoc <https://docs.gradle.org/current/javadoc/index.html?overview-summary.html>
- Groovy DSL <https://docs.gradle.org/current/dsl/index.html>

# User guide

Gradle Docs 5.3.1

Community ▾ Training News ▾ Enterprise

Search the docs

Docs Home

Tutorials

Release Notes

Gradle API

User Manual PDF

Getting Started

Installing Gradle

Upgrading Gradle...

Migrating to Gradle...

Troubleshooting Builds

Running Gradle Builds

Customizing Execution

Executing Multi-Project Builds

Inspecting Gradle Builds

Optimizing Build Times

Integrating Separate Gradle Builds (Composite Builds)

Authoring Gradle Builds

Learning the Basics

Organizing Build Logic

Authoring Maintainable Builds

Managing Dependencies

Publishing Artifacts

Java Projects

C++ Projects

Advanced Techniques

Example Gradle Projects

## Gradle User Manual

Gradle is an open-source build automation tool focused on flexibility and performance. Gradle build scripts are written using a [Groovy](#) or [Kotlin](#) DSL. Read about [Gradle features](#) to learn what is possible with Gradle.

Highly customizable — Gradle is modeled in a way that is customizable and extensible in the most fundamental ways.

Fast — Gradle completes tasks quickly by reusing outputs from previous executions, processing only inputs that changed, and executing tasks in parallel.

Powerful — Gradle is the official build tool for Android, and comes with support for many popular languages and technologies.



### New projects with Gradle

Getting started with Gradle is easy! First, follow our guide to [download and install Gradle](#), then check out Gradle [getting started guides](#) to create your first build.

If you're currently using Maven, see a visual [Gradle vs Maven comparison](#) and follow the guide for [migrating from Maven to Gradle](#).

### Using existing Gradle builds

Gradle supports many major IDEs, including Android Studio, Eclipse, IntelliJ IDEA, Visual Studio 2017, and XCode. You can also invoke Gradle via its [command-line interface](#) in your terminal or through your continuous integration server. [Gradle build scans](#) help you understand build results, improve build performance, and collaborate to fix problems faster.



Contents

New projects with Gradle

Using existing Gradle builds

Getting help

Licenses

## Gradle Tutorials and Guides

Here you can find project-based tutorials and topical guides to help you learn Gradle through using it. Whether you are new to Gradle or an experienced build master, the guides hosted here are designed to help you accomplish your goals.

### Getting Started

Step-by-step lessons on how to use Gradle, both in general and for specific tasks.

#### [Creating Build Scans](#)

Learn how to enable build scans for a Gradle build. Add the plugin and the license agreement, execute a scan, and view the results. Add build scan capabilities for all builds using an init script.

⌚ 8 mins

#### [Creating a New Gradle Build](#)

Learn Gradle tasks that work on any project, regardless of type. See what tasks are available, generate a wrapper, copy data from one location to another, add a plugin, and more.

⌚ 12 mins

Search



All Guides

Continuous Integration

Fundamentals

JVM

JavaScript

Native

Plugin Development

# Gradle guides

The screenshot shows the 'Writing Gradle Guides' page on the Gradle Guides website. The page has a sidebar on the left containing links to various sections like User Manual, Tutorials, DSL Reference, etc. The main content area includes sections for 'What you'll need' (with a bulleted list) and 'Create a GitHub issue' (with instructions and numbered steps). A sidebar on the right lists 'Contents' such as 'What you'll need', 'Create a GitHub issue', and 'Fork and clone the repository'. The top navigation bar includes links for Docs, Forums, Training, and Try Gradle Enterprise.

**Writing Gradle Guides**

This guide walks you through the process of writing a new guide for inclusion at the [Gradle Guides site](#). These guides can take different forms—such as short, step-by-step tutorials or longer, more discussion-oriented pieces—but the process for writing and contributing them is the same.

### What you'll need

- A [Git](#) client (command line is fine, or use one of the many excellent GUI clients)
- A [GitHub](#) account
- A [Java Runtime Environment](#), version 1.7 or better
- A text editor
- A basic familiarity with [AsciIDoc](#)

### Create a GitHub issue

The first step in creating a new Gradle guide is to submit a proposal to the [gradle/guides](#) GitHub repository. This is so that we, the Gradle Guides team, can work with you to determine the most appropriate subject, form and scope for the guide.

To submit your proposal:

1. Create a new issue at [gradle/guides](#) with a title that matches the suggested topic for the new guide. For example, "Publishing to private repositories" or "Creating fat JARs". The title should, in general, be a description of a task that a Gradle user might want to undertake.
2. Add a basic outline for the proposed guide, i.e. section headings

# Tooling

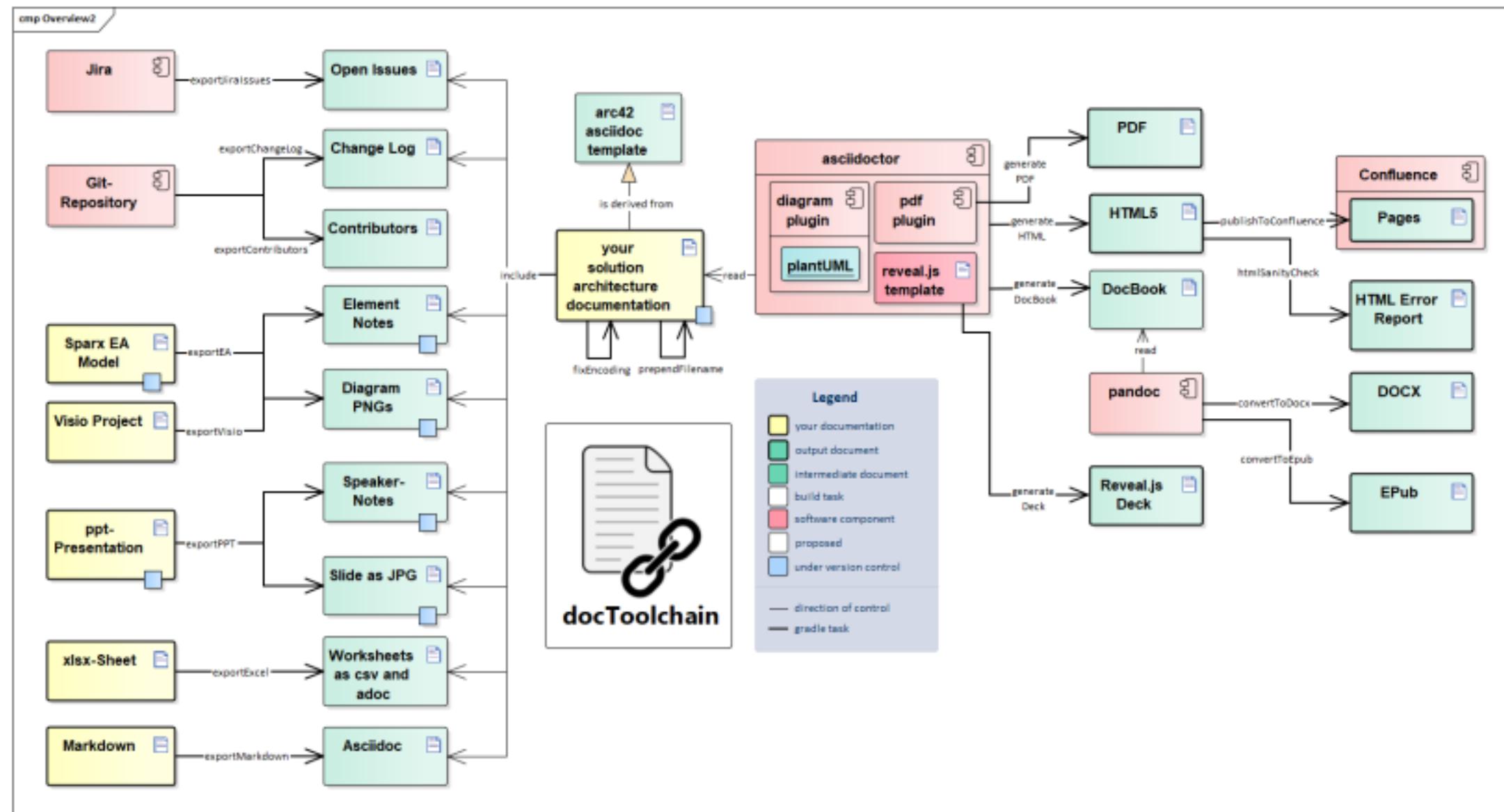
# Writing

- Plain Text
  - WYSIWYG Wiki Pages, Word Documents, Confluence, etc
- Markup
  - Asciidoc, Markdown, LaTeX
- Automated
  - Swagger / RAML [1]

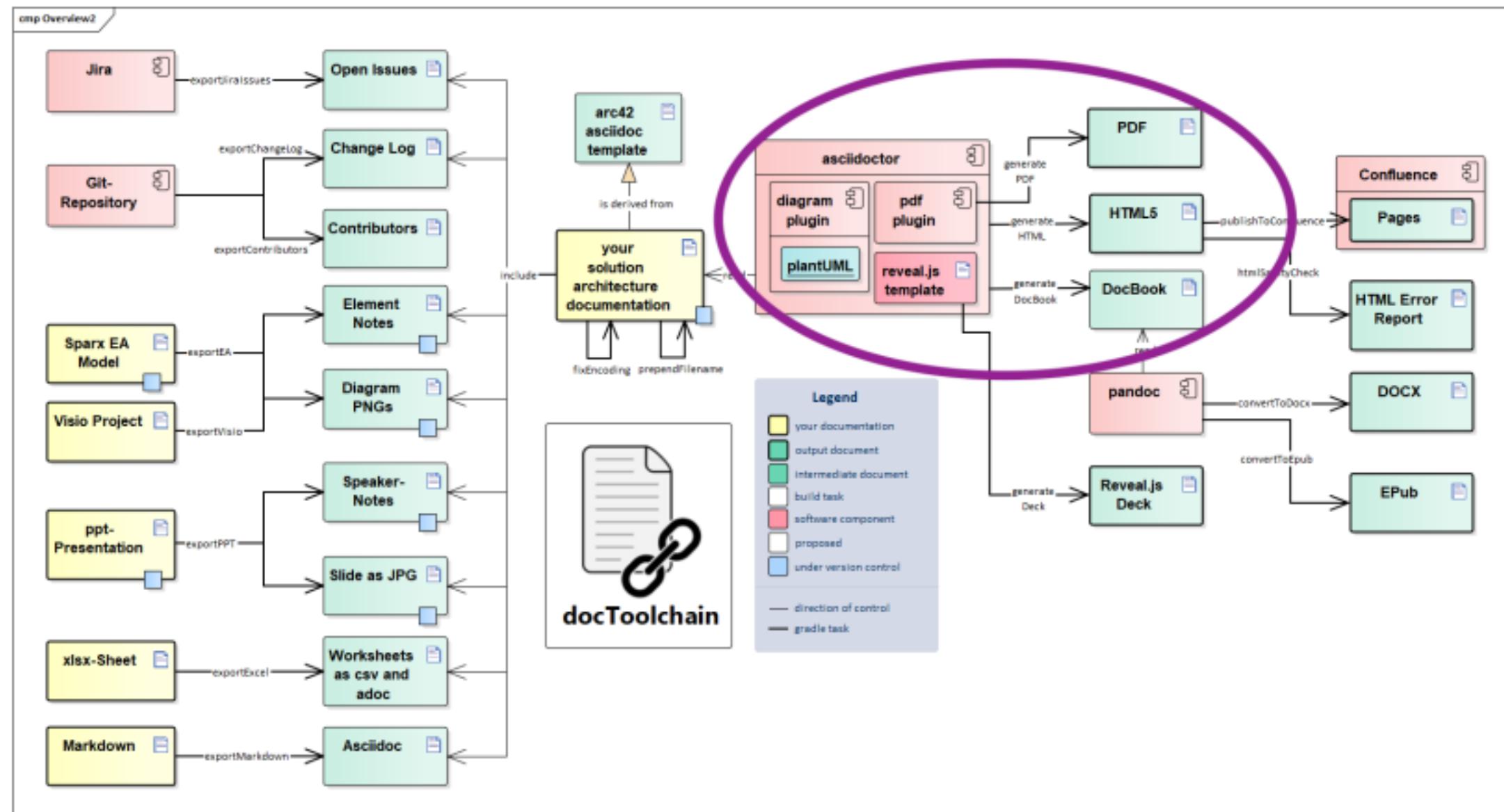
# Publishing

- Static sites
- PDFs
- Asciidoctor (for both ... and more!)

# Docs as Code



# Docs as Code



# Asciidoc (and Asciidoctor)

# What is Asciidoc?

- a format
- easily convertible to html, pdf, book formats, etc
- similar to markdown/latex
- recognized by GitHub as a readme format

# What is Asciidoctor?

- a tool to convert asciidoc files into other formats (called backends)
- written in ruby, but also has Gradle and Maven plugins
- easily extensible

[asciidoctor.org](http://asciidoctor.org)

# Let's Get Meta!

# Basic Format

```
:revnumber: {project-version}
:example-caption!:
:navigation:
:menu:
:status:
:title-slide-background-image: title.jpeg
:title-slide-transition: zoom
:title-slide-transition-speed: fast
:icons: font

= : Test-Driven Approaches to Documentation

+++<h2>+++
with Gradle, Asciidoctor, and Exemplar
+++</h2>+++
```

# Extensions

- Samples Macro
- Multi-Language Support

# Samples Macro

```
[source,groovy]
-----
include::sample[dir="groovy-dsl/code", files="build.gradle[tags=init]"
-----
```

as a GitHub gist

# Multi-Language Support



The screenshot shows a code editor interface with two tabs at the top: "Groovy" and "Kotlin". The "Groovy" tab is active. Below the tabs, there is a file named "build.gradle.kts". The code in the file is:

```
plugins {
    id("com.gradle.build-scan") version "2.1" ①
}
```

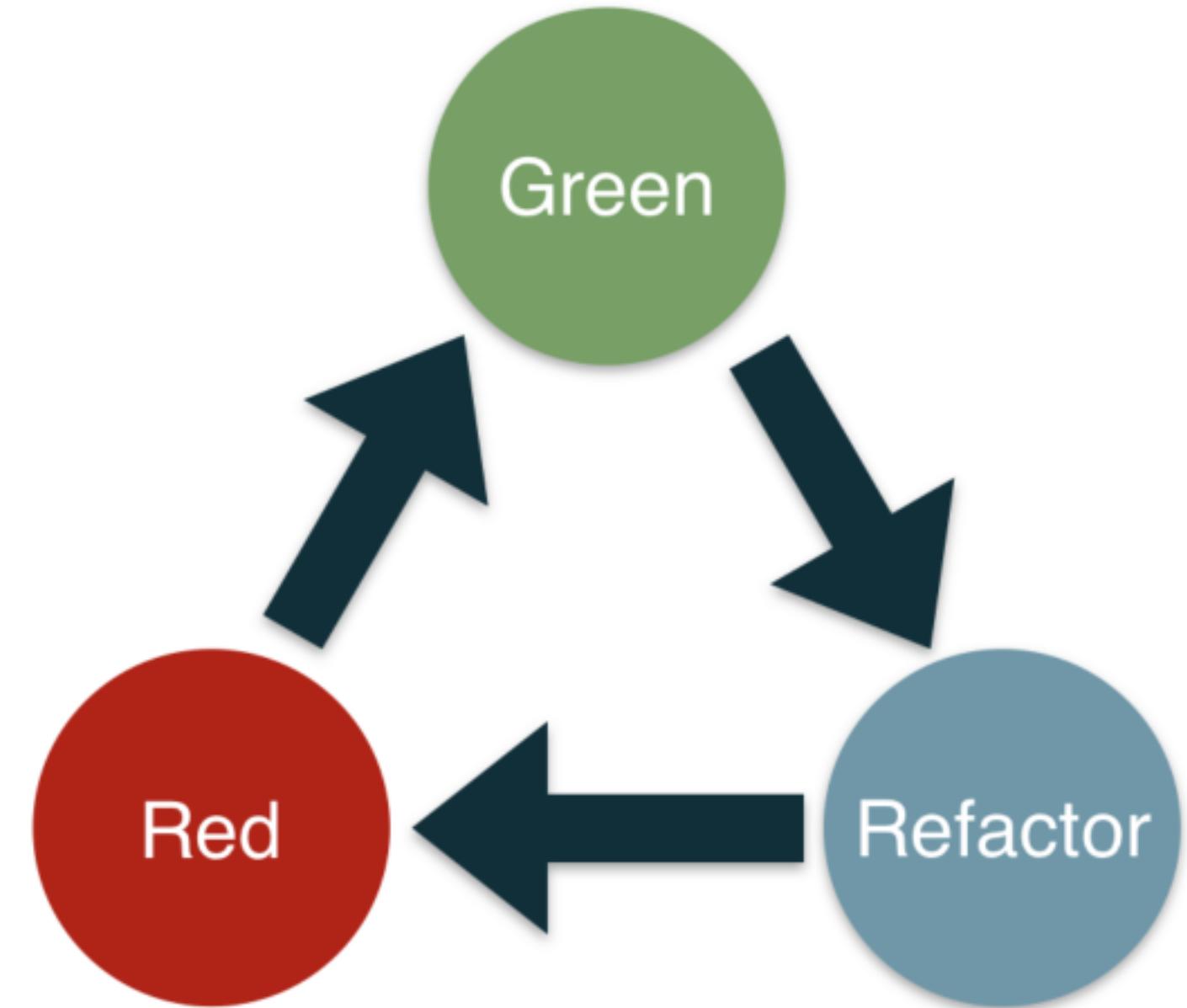
Below the code editor, there is a GitHub-style gist interface. It shows the beginning of a file with the following content:

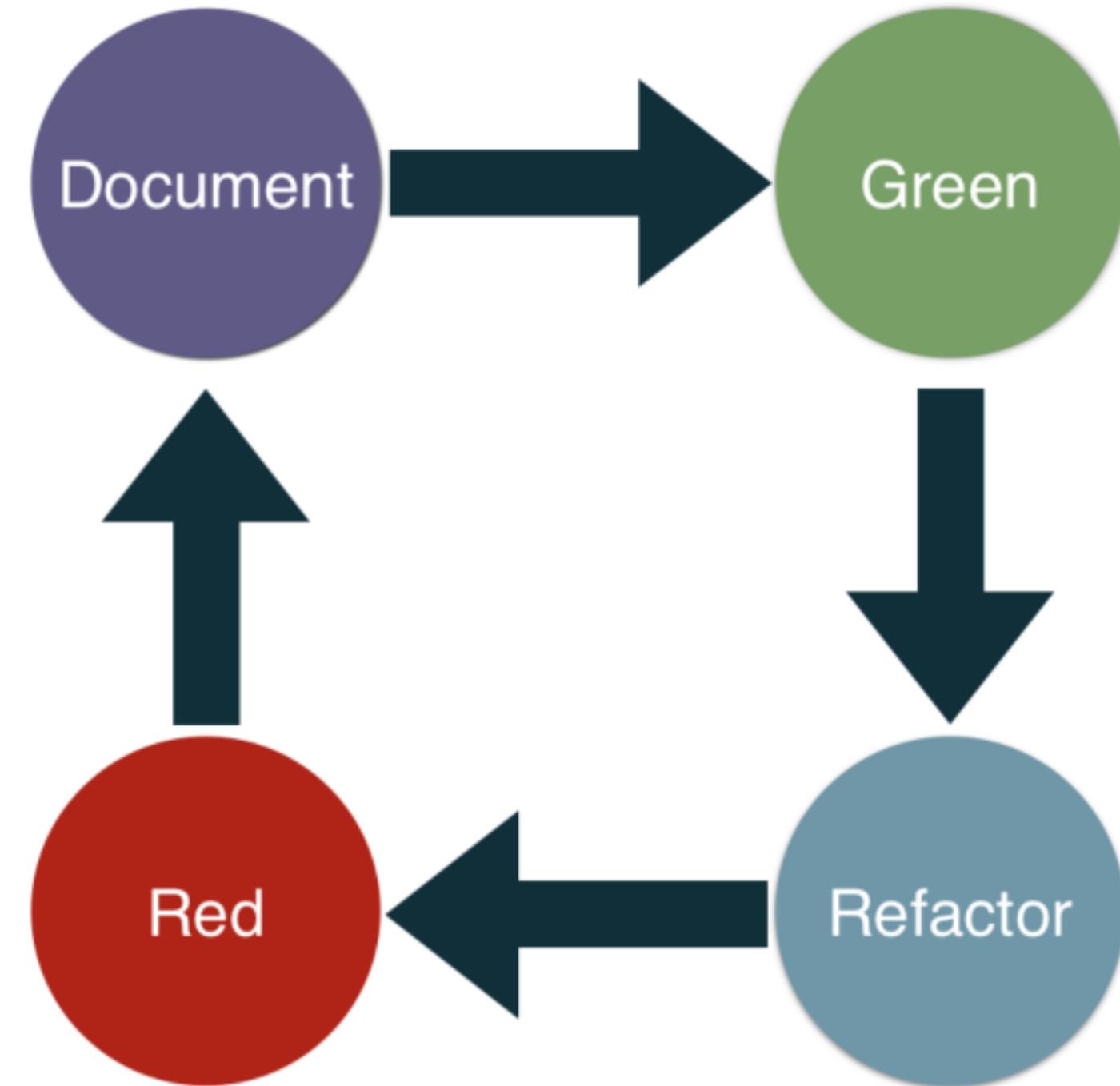
```
=====
include::sample[dir="groovy-dsl/code", files="build.gradle[tags=init]"
include::sample[dir="kotlin-dsl/code", files="build.gradle.kts[tags=ir
=====
```

At the bottom of the gist interface, there are navigation arrows and a progress bar.

as a GitHub gist

# Test Driven Documentation





# Introducing Exemplar

- Gradle/Exemplar on GitHub
- Finds samples, executes them, and verifies their outputs
- Project Lead
  - Eric Wendelin (@eriw), Head of Developer Experience at Gradle

# Quickstart

```
repositories {  
    maven {  
        url = uri("https://repo.gradle.org/gradle/libs")  
    }  
}  
  
dependencies {  
    testImplementation("org.gradle:sample-check:0.7.0")  
}
```

# Example Gradle Project

## build.gradle

```
plugins {  
    // Apply the java-library plugin to add support for Java Library  
    id 'java-library'  
}  
  
repositories {  
    // Use jcenter for resolving your dependencies.  
    // You can declare any Maven/Ivy/file repository here.  
    jcenter() ①  
}  
  
dependencies {  
    // This dependency is exported to consumers, that is to say found  
    api 'org.apache.commons:commons-math3:3.6.1'  
  
    // This dependency is used internally, and not exposed to consumers  
}
```

①

repository information

# Example Embedded Snippets

Sample title  
hello.groovy

```
println "Hello, World!"
```

```
$ groovy hello.groovy
Hello, World!
```

# Publishing

- make sure to run the tests

# Contributions Welcome

- **Idea:** Add markdown support
- help triage issues and PRs

[Gradle/Exemplar on GitHub](#)

# Other Testing Tools

# Gradle Test Kit

Test Kit

(mostly for Gradle Plugins)

## Gradle Guides Plugin

- Gradle Runner
  - Still present in some older guides
  - Good for cross-version testing
  - Really complex config
- CheckLinks
  - HEAD request to check all absolute links on the page are valid.

# Conclusion

- Documentation isn't always fun, but automation makes it easier!
- The only constant is change.

# Thank you!

# Talk to me!

- About Gradle
- About Gradle Enterprise
- About Careers at Gradle GmbH
- If you just want a Gradlephant sticker



# Resources

- Asciidoc
- Asciidoctor
- docToolchain
- Exemplar Documentation
- Gradle on GitHub