Offering a Stand-alone Mode for Your Spring Boot Library

Federico Mestrone

Software Engineer and Training Consultant

@fedmest www.federicomestrone.com

Library Overview

In this module:

- We will learn about Spring Profiles
  - To define groups of beans that add certain features to the core functionality
- We will offer a standalone mode
  - With the explorer website we have been using so far
  - With a command line interface that can be run in a shell
  - Either can be selected individually with a dedicated profile

Spring Profiles allow you to register different beans in different scenarios

Environment: Dev vs. Stage vs. Prod

Specialized Optional Functionality

Alternative Implementations

Customer Needs

Profile Definitions

```
@Component
@Profile("profile1")
public class MyFeatureBean {

    // ...

}
```

Using Profiles

Including based on a profile being selected

```
@Component
@Profile("!profile1")
public class MyFeatureBean {

    // ...

}
```

Using Profiles

Including based on a profile NOT being selected

```
@Component
@Profile({"profile1","profile2"})
public class MyFeatureBean {

    // ...

}
```

Using Profiles

Including based on any of several profiles being selected

```
@Component
@Profile("profile1 | profile2")
public class MyFeatureBean {

    // ...

}
```

Using Profiles

Including based on any of several profiles being selected, but using the expression syntax

```
@Component
@Profile("profile1 & profile2")
public class MyFeatureBean {

    // ...

}
```

Using Profiles

Including based on all given profiles being selected at the same time

Profile Selection Expressions

&	**Logical AND: both profiles must be active**
|	**Logica OR: only one profile needs to be active**
!	**Logical NOT: the profile must not be active**
( )	**Must be used when having mixing & and | in the same expression**

Profile-based Configuration

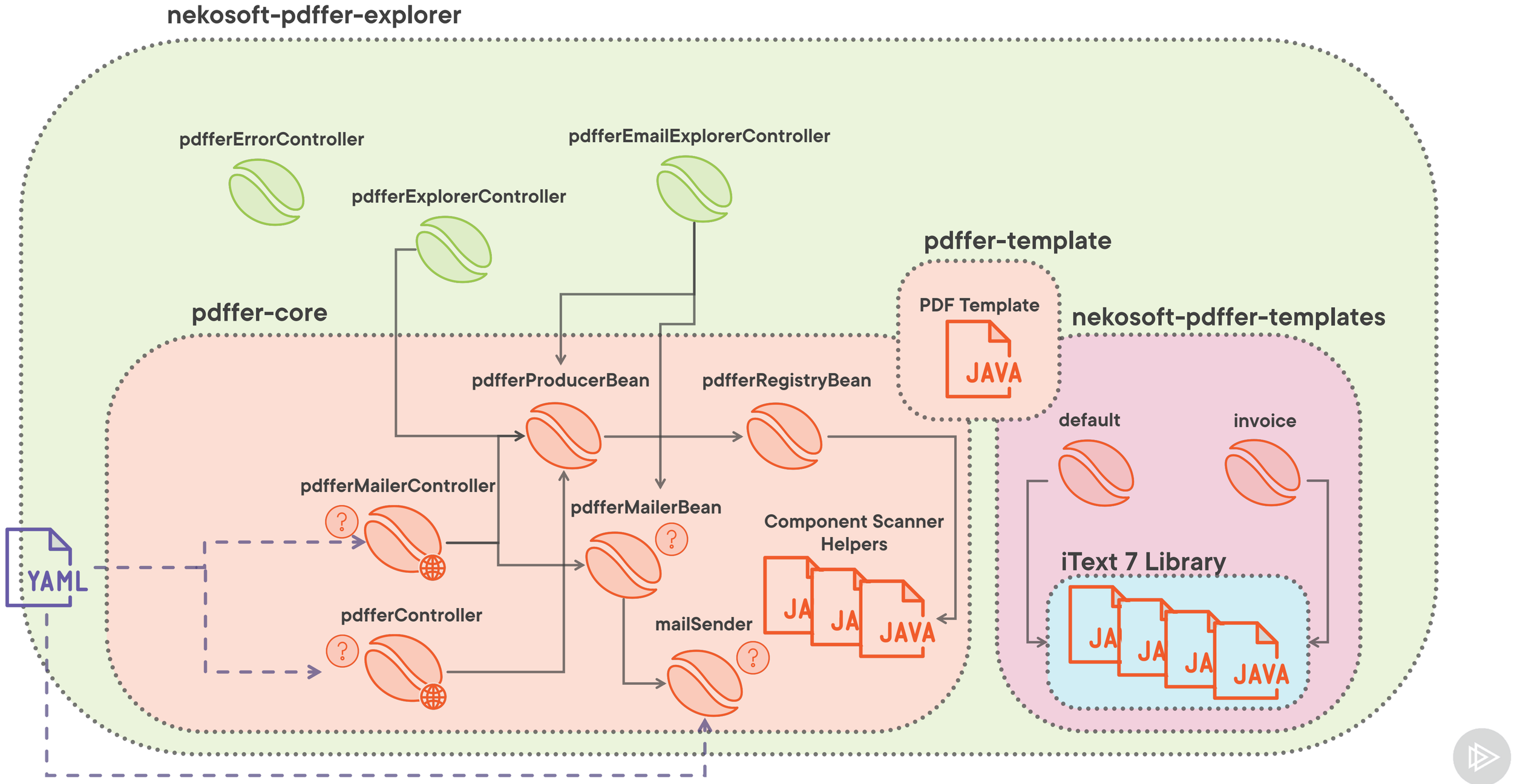
You can apply profiles to full @Configuration classes

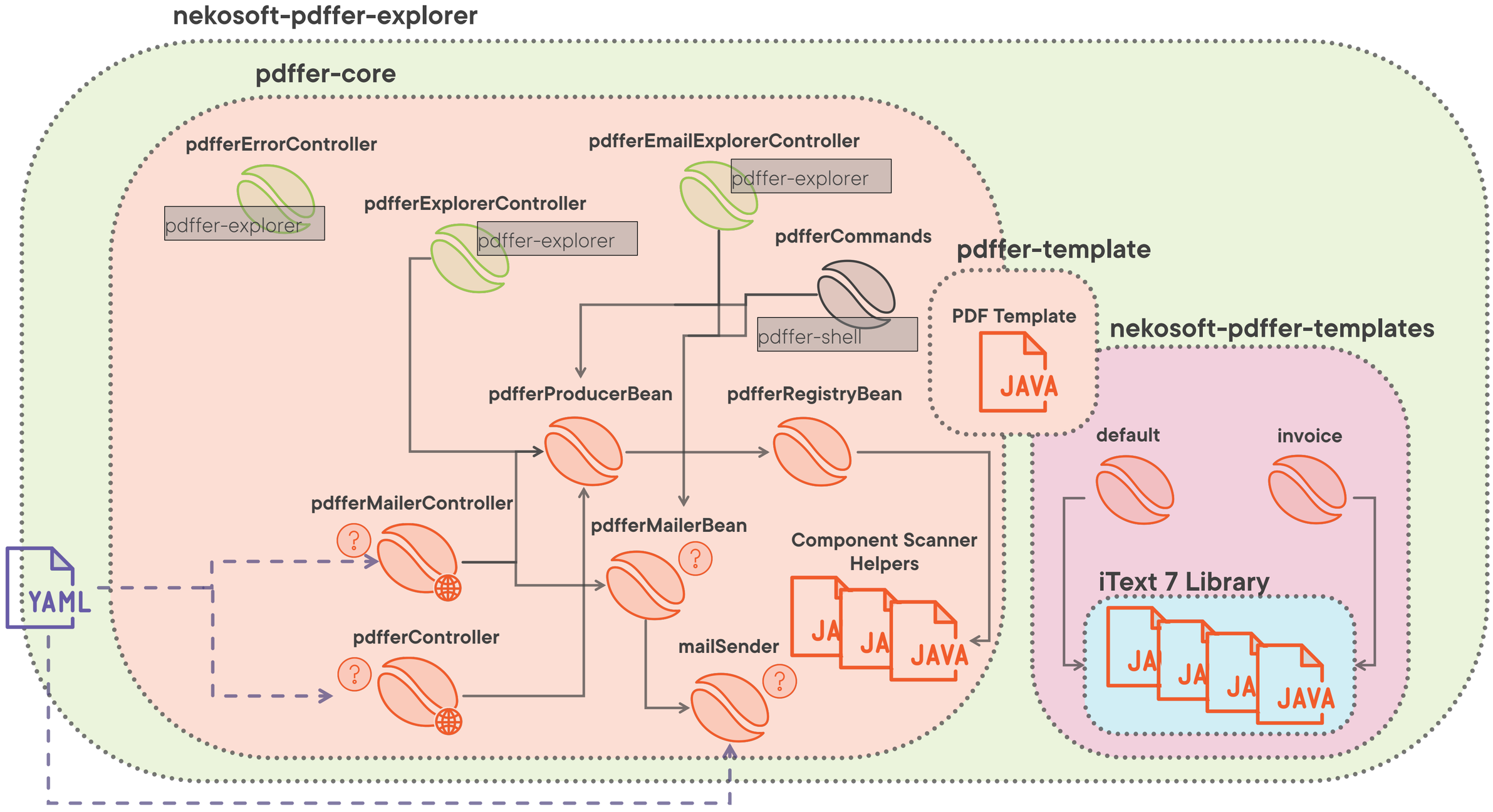
Development environment

```
@Configuration
@Profile("development")
public class DevConnPoolConfig {
    @Bean
    public DataSource dataPool1() {
        // dev database connection pool
    }
    @Bean
    public DataSource dataPool2() {
        // dev database connection pool
    }
}
```

Production environment

```
@Configuration
@Profile("production")
public class ProdConnPoolConfig {
    @Bean
    public DataSource dataPool1() {
        // prod database connection pool
    }
    @Bean
    public DataSource dataPool2() {
        // prod database connection pool
    }
}
```


nekosoft-pdffer-explorer
pdfferErrorController
pdfferEmailExplorerController
pdfferExplorerController
pdffer-template
pdffer-core
PDF Template
JAVA
nekosoft-pdffer-templates
pdfferProducerBean
pdfferRegistryBean
default
invoice
pdfferMailerController
pdfferMailerBean
Component Scanner
Helpers
YAML
iText 7 Library
pdfferController
mailSender
JA
JAVA


nekosoft-pdffer-explorer
pdffer-core
pdfferErrorController
pdffer-explorer
pdfferExplorerController
pdffer-explorer
pdfferEmailExplorerController
pdffer-explorer
pdfferCommands
pdffer-shell
pdffer-template
PDF Template
JAVA
nekosoft-pdffer-templates
default
invoice
iText 7 Library
JA
JA
JA
JAVA
pdfferProducerBean
pdfferRegistryBean
pdfferMailerController
pdfferMailerBean
Component Scanner
Helpers
JA
JA
JAVA
YAML
pdfferController
mailSender

Spring Shell

Offers a full-feature shell that supports the creation of custom commands

Adding Shell Dependency (Maven)

```
<dependencies>
    <dependency>
        <groupId>org.springframework.boot</groupId>
        <artifactId>spring-boot-starter</artifactId>
    </dependency>
    <dependency>
        <groupId>org.springframework.shell</groupId>
        <artifactId>spring-shell-starter</artifactId>
        <version>2.0.1.RELEASE</version>
        <optional>true</optional>
    </dependency>
    <!-- ... -->
</dependencies>
```

Adding Shell Dependency (Gradle)

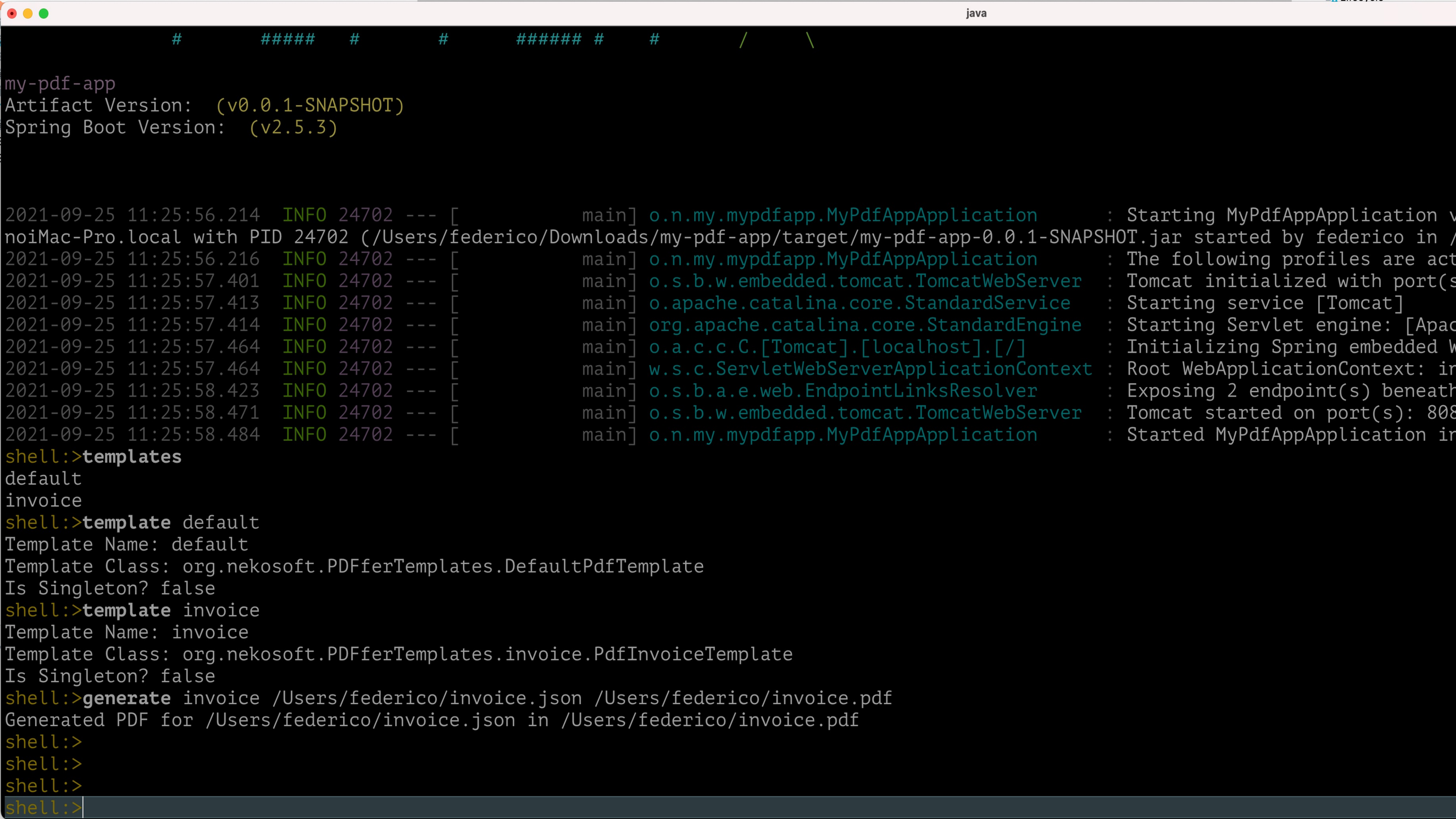
```
java {
    registerFeature('web') {
        usingSourceSet(sourceSets.main)
    }
    registerFeature('email') {
        usingSourceSet(sourceSets.main)
    }
    registerFeature('shell') {
        usingSourceSet(sourceSets.main)
    }
}
```

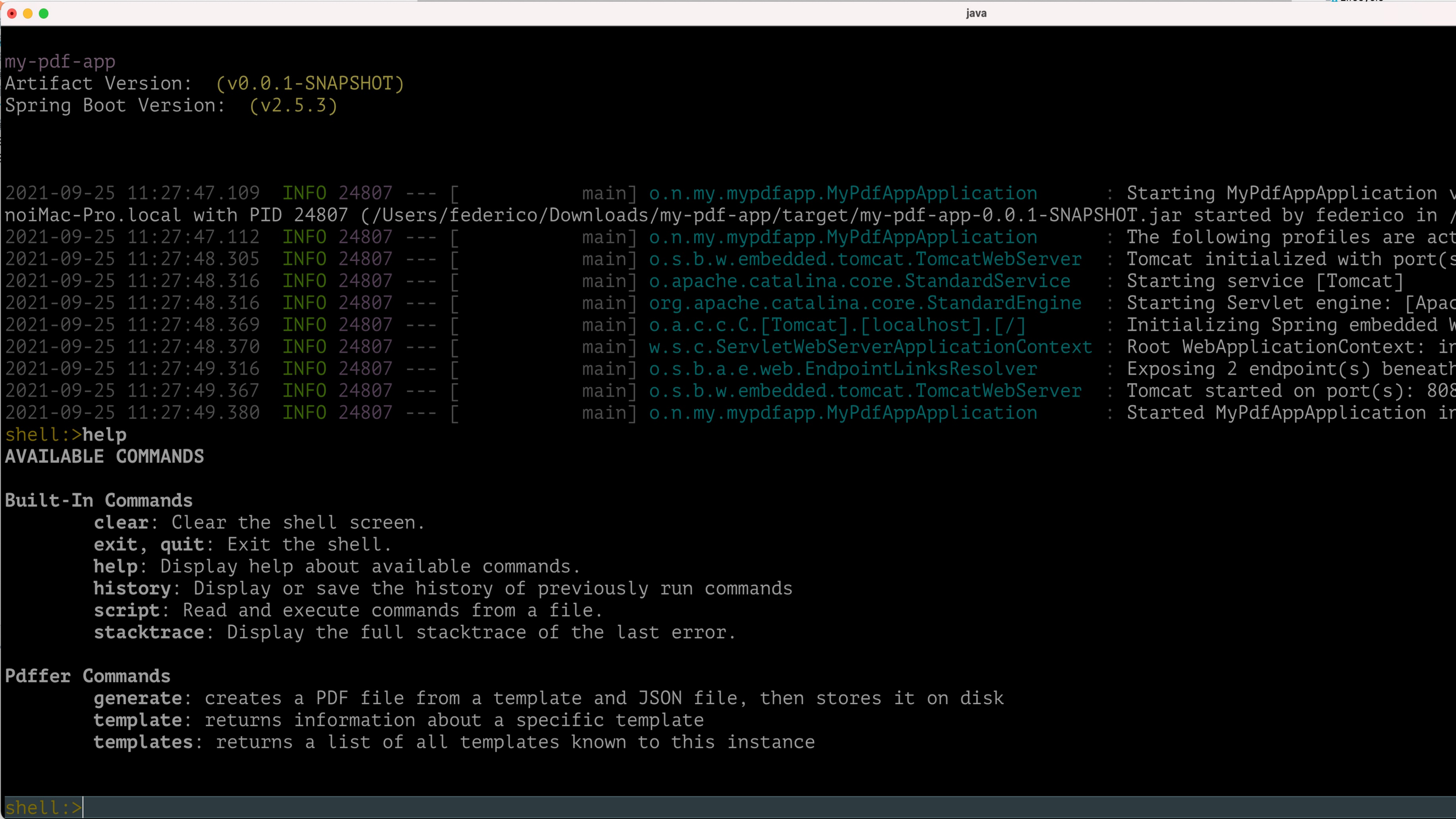
Adding Shell Dependency (Gradle)

```
dependencies {
    implementation 'org.springframework.boot:spring-boot-starter'
    api(project(':pdffer-template'))
    webApi 'org.springframework.boot:spring-boot-starter-web'
    emailApi 'org.springframework.boot:spring-boot-starter-mail'
    shellApi 'org.springframework.shell:spring-shell-starter:2.0.1.RELEASE'
    developmentOnly 'org.springframework.boot:spring-boot-devtools'
    annotationProcessor "org.springframework.boot:spring-boot-configuration-processor"
}
```

Adding Shell Dependency (Gradle)

```
@ShellComponent
@ConditionalOnClass(name = "org.springframework.shell.standard.ShellComponent")
@Profile("pdffer-shell")
public class PdfferCommands {

    public PdfferCommands(PdfferRegistryBean registry, PdfferProducerBean producer)

    @ShellMethod("returns a list of all templates known to this instance")
    public List<String> templates() { /* ... */ }

    @ShellMethod("returns information about a specific template")
    public List<String> template(String name) { /* ... */ }

    @ShellMethod("creates PDF file from template + JSON file, then stores it on disk")
    public void generate(String template, String input, String output) { /* ... */ }
}
```


java
/ \
my-pdf-app
Artifact Version: (v0.0.1-SNAPSHOT)
Spring Boot Version: (v2.5.3)
2021-09-25 11:25:56.214 INFO 24702 --- [main] o.n.my.mypdfapp.MyPdfAppApplication : Starting MyPdfAppApplication v
noiMac-Pro.local with PID 24702 (/Users/federico/Downloads/my-pdf-app/target/my-pdf-app-0.0.1-SNAPSHOT.jar started by federico in /
2021-09-25 11:25:56.216 INFO 24702 --- [main] o.n.my.mypdfapp.MyPdfAppApplication : The following profiles are act
2021-09-25 11:25:57.401 INFO 24702 --- [main] o.s.b.w.embedded.tomcat.TomcatWebServer : Tomcat initialized with port(s
2021-09-25 11:25:57.413 INFO 24702 --- [main] o.apache.catalina.core.StandardService : Starting service [Tomcat]
2021-09-25 11:25:57.414 INFO 24702 --- [main] org.apache.catalina.core.StandardEngine : Starting Servlet engine: [Apac
2021-09-25 11:25:57.464 INFO 24702 --- [main] o.a.c.c.C.[Tomcat].[localhost].[/] : Initializing Spring embedded W
2021-09-25 11:25:57.464 INFO 24702 --- [main] w.s.c.ServletWebServerApplicationContext : Root WebApplicationContext: in
2021-09-25 11:25:58.423 INFO 24702 --- [main] o.s.b.a.e.web.EndpointLinksResolver : Exposing 2 endpoint(s) beneath
2021-09-25 11:25:58.471 INFO 24702 --- [main] o.s.b.w.embedded.tomcat.TomcatWebServer : Tomcat started on port(s): 808
2021-09-25 11:25:58.484 INFO 24702 --- [main] o.n.my.mypdfapp.MyPdfAppApplication : Started MyPdfAppApplication in
shell:>templates
default
invoice
shell:>template default
Template Name: default
Template Class: org.nekosoft.PDFferTemplates.DefaultPdfTemplate
Is Singleton? false
shell:>template invoice
Template Name: invoice
Template Class: org.nekosoft.PDFferTemplates.invoice.PdfInvoiceTemplate
Is Singleton? false
shell:>generate invoice /Users/federico/invoice.json /Users/federico/invoice.pdf
Generated PDF for /Users/federico/invoice.json in /Users/federico/invoice.pdf
shell:>
shell:>
shell:>
shell:>


java
my-pdf-app
Artifact Version: (v0.0.1-SNAPSHOT)
Spring Boot Version: (v2.5.3)
2021-09-25 11:27:47.109 INFO 24807 --- [main] o.n.my.mypdfapp.MyPdfAppApplication : Starting MyPdfAppApplication v
noiMac-Pro.local with PID 24807 (/Users/federico/Downloads/my-pdf-app/target/my-pdf-app-0.0.1-SNAPSHOT.jar started by federico in /
2021-09-25 11:27:47.112 INFO 24807 --- [main] o.n.my.mypdfapp.MyPdfAppApplication : The following profiles are act
2021-09-25 11:27:48.305 INFO 24807 --- [main] o.s.b.w.embedded.tomcat.TomcatWebServer : Tomcat initialized with port(s
2021-09-25 11:27:48.316 INFO 24807 --- [main] o.apache.catalina.core.StandardService : Starting service [Tomcat]
2021-09-25 11:27:48.316 INFO 24807 --- [main] org.apache.catalina.core.StandardEngine : Starting Servlet engine: [Apac
2021-09-25 11:27:48.369 INFO 24807 --- [main] o.a.c.c.C.[Tomcat].[localhost].[/] : Initializing Spring embedded W
2021-09-25 11:27:48.370 INFO 24807 --- [main] w.s.c.ServletWebServerApplicationContext : Root WebApplicationContext: in
2021-09-25 11:27:49.316 INFO 24807 --- [main] o.s.b.a.e.web.EndpointLinksResolver : Exposing 2 endpoint(s) beneath
2021-09-25 11:27:49.367 INFO 24807 --- [main] o.s.b.w.embedded.tomcat.TomcatWebServer : Tomcat started on port(s): 808
2021-09-25 11:27:49.380 INFO 24807 --- [main] o.n.my.mypdfapp.MyPdfAppApplication : Started MyPdfAppApplication in
shell:>help
AVAILABLE COMMANDS
Built-In Commands
clear: Clear the shell screen.
exit, quit: Exit the shell.
help: Display help about available commands.
history: Display or save the history of previously run commands
script: Read and execute commands from a file.
stacktrace: Display the full stacktrace of the last error.
Pdffer Commands
generate: creates a PDF file from a template and JSON file, then stores it on disk
template: returns information about a specific template
templates: returns a list of all templates known to this instance
shell:>

```
        help: Display help about available commands.
        history: Display or save the history of previously run commands
        script: Read and execute commands from a file.
        stacktrace: Display the full stacktrace of the last error.

Pdffer Commands
        generate: creates a PDF file from a template and JSON file, then stores it on disk
        template: returns information about a specific template
        templates: returns a list of all templates known to this instance


shell:>help generate


NAME
        generate - creates a PDF file from a template and JSON file, then stores it on disk

SYNOPSYS
        generate [--template] string  [--input] string  [--output] string

OPTIONS
        --template  string

                [Mandatory]

        --input  string

                [Mandatory]

        --output  string

                [Mandatory]


shell:>
```

Profile Activation

Beans and configurations can be associated to a profile

- Only created if the profile is active

More than one profile can be active at any time

- Beans from all active profiles will be included

Several ways to select active profiles

Activating a Profile in Spring

ServletContext

spring.profiles.active
init parameter

ConfigurableEnvironment

setActiveProfiles
method

JVM System Parameter

-Dspring.profiles.active

Environment Variable

SPRING_PROFILES_ACTIVE

Default Profile

spring.profiles.default

Spring Boot Options

SpringApplication Class

setAdditionalProfiles **method**

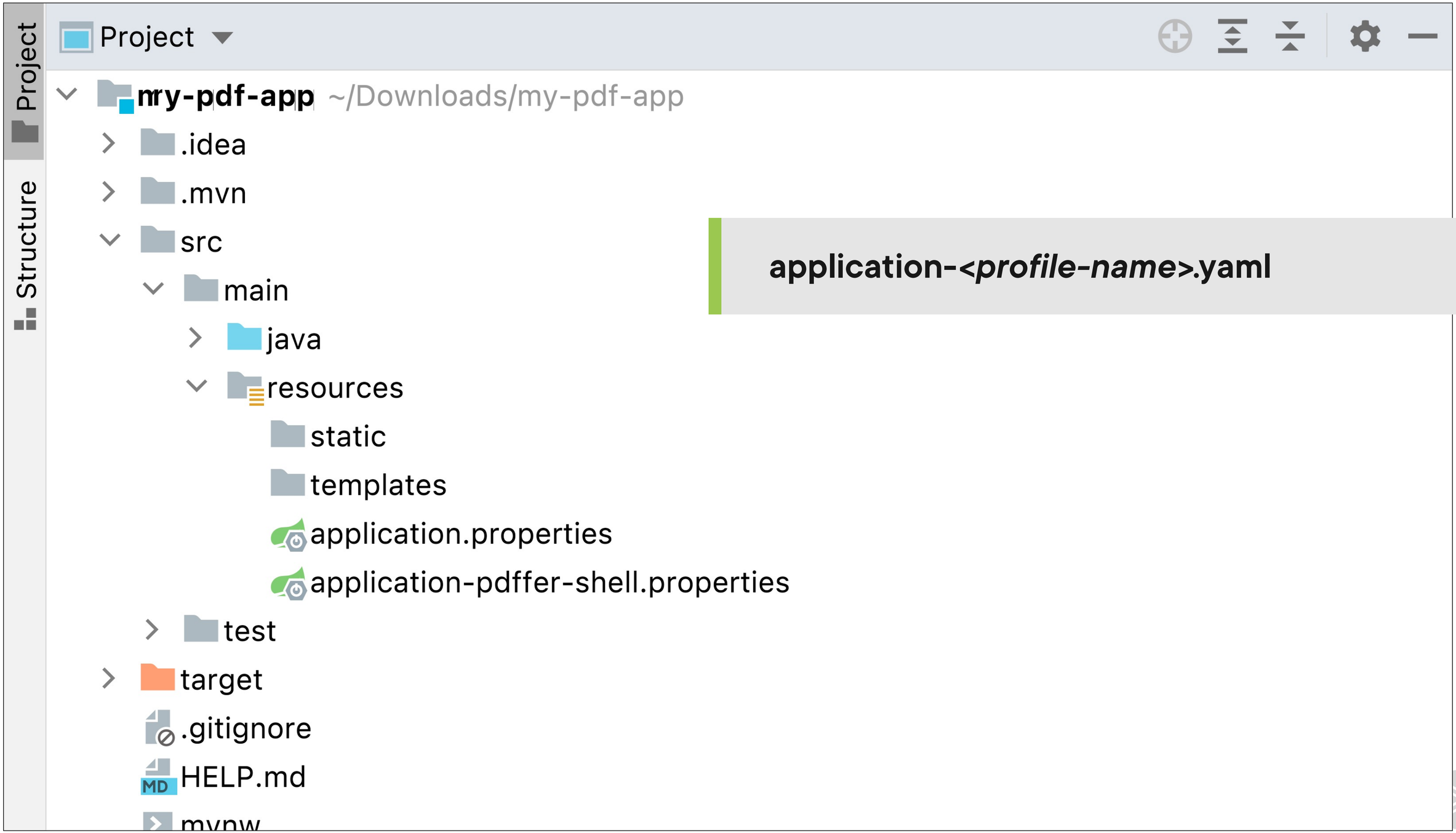
Standard Property in *application.yaml*

spring.profiles.active

Profile-based Configuration Properties


Default properties of Spring application
(set in code)
@PropertySource
(in @Configuration classes)
application.properties outside of JAR
application.properties packaged in JAR
RandomValueProperty Source
(random.*)
OS environment variables
Java system properties
(set in code or command line)
SPRING_APPLICATION_JSON
ServletConfig
ServletContext
JNDI Attributes
Command line arguments to the application
Devtools global settings properties
@TestPropertySource
Properties in tests


Default properties of Spring application
(set in code)
@PropertySource
(in @Configuration classes)
profile-specific application.properties outside of JAR
application.properties outside of JAR
profile-specific application.properties packaged in JAR
application.properties packaged in JAR
RandomValueProperty Source
(random.*)
OS environment variables
Java system properties
(set in code or command line)
SPRING_APPLICATION_JSON
ServletConfig
ServletContext
JNDI Attributes
Command line arguments to the application
Devtools global settings properties
@TestPropertySource
Properties in tests


Project
Structure
Project
my-pdf-app ~/Downloads/my-pdf-app
.idea
.mvn
src
main
java
resources
static
templates
application.properties
application-pdffer-shell.properties
test
target
.gitignore
HELP.md
mvnw
application-<profile-name>.yaml

One YAML File, Multiple YAML Documents

```yaml
dbcaching:
  refresh:
    rate: 0
---
spring:
  config:
    activate:
      on-profile: "staging"
dbcaching:
  refresh:
    rate: 5
---
spring:
  config:
    activate:
      on-profile: "production"
dbcaching:
  refresh:
    rate: 10
```

Profiles in Maven and Spring

Maven profiles apply at build-time

Spring profiles apply at runtime

Synchronizing them means you get

- profile-based dependencies and other build-time properties
- profile-based Spring beans and other run-time properties

Changing profile requires a rebuild

- tool for generating different products
- not for making one product more flexible

Summary

Effective development

- Spring Initialzr
  - From the web, console, or IDE
- Navigating Spring configurations in IDE
- Testing HTTP endpoints
- Externalizing configuration
  - With property POJOs
  - And configuration metadata
- Using Spring Profiles
  - for bean selection
  - for property value selection

Summary

Effective configuration

- Spring Boot autoconfiguration
  - The spring.factories file
- Hierarchical contexts
- Advanced component scanning
  - And custom type filters
- Built-in and custom conditional beans
- Spring Boot configuration properties
- Conditionals on property values
- How to activate profiles
  - Multi-document YAML files
- Profile groups

Summary

Effective deployment

- Spring Dev tools
  - Auto-reload of source code changes
- Logging settings in Spring Boot
  - Conditions Evaluation Report
- Optional dependencies in Maven
  - And features in Gradle
- CLI with Spring Shell
- Spring profiles and Maven profiles

Up Next:
Deploying your Spring Boot Applications