

Spring Shell: создаём консольное приложение

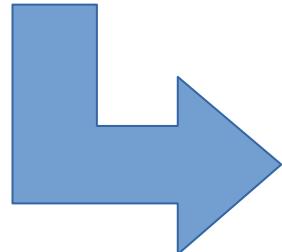
Каникулы!

Что такое Spring Shell?

Сайт проекта: <http://projects.spring.io/spring-shell/>

С помощью Spring Shell можно быстро создавать консольные приложения просто добавив специальные аннотации к методам.

```
@ShellComponent  
@SpringBootApplication  
open class ConsoleMessenger {  
  
    @ShellMethod( value: "Register user")  
    fun register(login: String, password: String) : User {  
        if (Storage.users.any { it.login == login }) {  
            throw UserAlreadyExistsException()  
        }  
    }  
}
```



```
.\bin\spring-shell-1.5.3.RELEASE.jar  
=====|=====  
:: Spring Boot ::      (v1.5.3.RELEASE)  
shell:>  
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.  
  
Console Messenger  
  add-contact: Add contact  
  contacts: ListContacts  
  login: Login  
  logout: Logout  
  register: Register user  
  remove: Remove user  
  
shell:>
```

build.gradle

```
buildscript {  
    ext.kotlin_version = '1.2.71' // Required for Kotlin integration  
    ext.spring_boot_version = '1.5.3.RELEASE'  
    repositories {  
        jcenter()  
        mavenLocal()  
    }  
    dependencies {  
        classpath "org.jetbrains.kotlin:kotlin-gradle-plugin:$kotlin_version" // Required for Kotlin integration  
        classpath "org.springframework.boot:spring-boot-gradle-plugin:$spring_boot_version"  
    }  
}  
  
plugins {  
    id 'org.jetbrains.kotlin.jvm' version '1.2.51'  
}  
  
apply plugin: 'kotlin' // Required for Kotlin integration  
apply plugin: 'org.springframework.boot'  
apply plugin: 'application'  
  
mainClassName = 'ru.senin.sample.ConsoleMessengerKt'  
  
group 'ru.senin.kotlin'  
version '1.0-SNAPSHOT'  
  
repositories {  
    mavenCentral()  
    maven {  
        url 'https://repo.spring.io/libs-snapshot'  
    }  
}  
dependencies {  
    compile "org.jetbrains.kotlin:kotlin-stdlib-jdk8"  
    compile "org.jetbrains.kotlin:kotlin-reflect"  
    compile 'org.springframework.shell:spring-shell-starter:2.0.1.BUILD-SNAPSHOT'  
    testCompile 'org.junit.jupiter:junit-jupiter-api:5.0.2'  
}  
compileKotlin {  
    kotlinOptions.jvmTarget = "1.8"  
}  
compileTestKotlin {  
    kotlinOptions.jvmTarget = "1.8"  
}
```

Аннотации в коде

```
package ru.senin.sample

import org.springframework.boot.SpringApplication
import org.springframework.boot.autoconfigure.SpringBootApplication
import org.springframework.shell.standard.ShellComponent
import org.springframework.shell.standard.ShellMethod
import java.util.*
import kotlin.collections.ArrayList
import kotlin.collections.HashMap

fun main(args: Array<String>) {
    // запускаем приложение Spring
    SpringApplication.run(ConsoleMessenger::class.java, *args)
}

// Аннотация, указывающая, что данный класс является приложением Spring.
// Всё чуть сложнее, но пока не будем вдаваться в подробности :)
@SpringBootApplication
// Аннотация, указывающая, что в данном классе определяются консольные команды
@ShellComponent
open class ConsoleMessenger { // класс должен быть open, чтобы сработала магия Spring!

    // Методы класса, которые хотим превратить в команды помечаем аннотацией @ShellMethod
    @ShellMethod("Register user")
    fun register(login: String, password: String) : User {
        if (Storage.users.any { it.login == login }) {
            throw UserAlreadyExistsException()
        }
        val newUser = User(login, password)
        Storage.users.add(newUser)
        return newUser
    }
}
```

Спасибо за внимание!

Сенин Михаил Андреевич

Email: mikhail.senin@jetbrains.com

<https://vk.com/kotlin566>