working with yaml file for spring boot profiles
==========

application.yml
==========
```
spring:
  profiles:
    active: dev
```

application-dev.yml
==========
```
spring:
  datasource:
    driver-class-name: com.mysql.cj.jdbc.Driver
    password: root123
    url: jdbc:mysql:///enterprisejavabatch
    username: root
```

application-test.yml
==========
```
spring:
  datasource:
    driver-class-name: com.mysql.cj.jdbc.Driver
    password: root123
    type: org.apache.commons.dbcp2.BasicDataSource
    url: jdbc:mysql:///enterprisejavabatch
    username: root
```

application-uat.yml
==========
```
spring:
  datasource:
    driver-class-name: oracle.jdbc.driver.OracleDriver
    password: root123
    url: jdbc:oracle:thin:@localhost:1521:XE
    username: System
```

application-prod.yml
==========
```
spring:
  datasource:
    driver-class-name: oracle.jdbc.driver.OracleDriver
    password: root123
    url: jdbc:oracle:thin:@localhost:1521:XE
    username: System
```

pom.xml
==========
```
<dependencies>
      <dependency>
            <groupId>org.springframework.boot</groupId>
            <artifactId>spring-boot-starter-jdbc</artifactId>
      </dependency>
      <dependency>
            <groupId>com.mysql</groupId>
            <artifactId>mysql-connector-j</artifactId>
            <scope>runtime</scope>
      </dependency>
```

```
	<dependency>
		<groupId>com.oracle.database.jdbc</groupId>
		<artifactId>ojdbc8</artifactId>
		<scope>runtime</scope>
	</dependency>

	<!-- https://mvnrepository.com/artifact/com.oracle.database.jdbc/ucp -->
	<dependency>
		<groupId>com.oracle.database.jdbc</groupId>
		<artifactId>ucp</artifactId>
	</dependency>

	<!-- https://mvnrepository.com/artifact/org.apache.commons/commons-dbcp2 -->
	<dependency>
		<groupId>org.apache.commons</groupId>
		<artifactId>commons-dbcp2</artifactId>
	</dependency>

	<!-- https://mvnrepository.com/artifact/com.mchange/c3p0 -->
	<dependency>
		<groupId>com.mchange</groupId>
		<artifactId>c3p0</artifactId>
		<version>0.9.5.5</version>
	</dependency>
</dependencies>


@SpringBootApplication
public class BootProj11RealTimeDiAutoConfigurationProfilesApplication {
	@Autowired
	private Environment env;

	@Bean(name = "dataSource")
	@Profile({"dev","uat"})
	public ComboPooledDataSource createDS() throws Exception {


System.out.println("BootProj11RealTimeDiAutoConfigurationProfilesApplication.create
DS()");
		ComboPooledDataSource dataSource = new ComboPooledDataSource();
		dataSource.setJdbcUrl(env.getProperty("spring.datasource.url"));
		dataSource.setUser(env.getProperty("spring.datasource.username"));
		dataSource.setPassword(env.getProperty("spring.datasource.password"));
		return dataSource;
	}
}

working with single yaml file
==============================
spring:
  datasource:
    driver-class-name: com.mysql.cj.jdbc.Driver
    password: root123
    url: jdbc:mysql:///enterprisejavabatch
    username: root
  profiles:
    active: dev

---
```

```
spring:
  datasource:
    driver-class-name: com.mysql.cj.jdbc.Driver
    password: root123
    url: jdbc:mysql:///enterprisejavabatch
    username: root
  config:
    activate:
      on-profile: dev
---
spring:
  datasource:
    driver-class-name: oracle.jdbc.driver.OracleDriver
    password: root123
    url: jdbc:oracle:thin:@localhost:1521:XE
    username: System
  config:
    activate:
      on-profile: test
---
spring:
  datasource:
    driver-class-name: com.mysql.cj.jdbc.Driver
    password: root123
    type: org.apache.commons.dbcp2.BasicDataSource
    url: jdbc:mysql:///enterprisejavabatch
    username: root
  config:
    activate:
      on-profile: uat
---

spring:
  datasource:
    driver-class-name: oracle.jdbc.driver.OracleDriver
    password: root123
    url: jdbc:oracle:thin:@localhost:1521:XE
    username: System
  config:
    activate:
      on-profile: prod
```

Runners in springboot
==========
=> Runners are java classes which would act like Spring bean of SpringBoot which are implementin XxxxRunners directly or indirectly.
=> Every runner class contains run() dealing with onetime executing logic and those logics will execute where SpringApplication.run()
   is about to complete all its startup activities(right after creating a ApplicationContext object and completing
   preinstantiation and injections)
=> The run() of every Runner class will be executed automatically by IOC container only for one time as a part of ApplicationStartup
   process that takes place in SpringApplication.run() method.

There are 2 types of Runners in SpringBoot
a. CommandLineRunner(I)
b. ApplicationRunner(I)

Note: Both the runner are giving run() as the callback method[becoz it get called by underlying IOC container automatically]
      Both the runner run() method gets the command line arguments as the parameter values but the way they get them is
       different.

Saturday
========
SpringJDBC,SpringORM,SpringDataJPA

Sunday
======
SpringBootMongoDB(sunday)
SpringMVC(sunday)