Java Secrets, API, 8+

Шестая лекция

Secrets storing

- Constants
- Database
- Properties file
- Environment variables
- Environment + Properties file

Constants

- Pros
  - Easy to write
  - Easy to use in code
- Cons
  - Everything else

```
public class VerySecureConfigs {
  public static final String URL = "sberbank.ru";
  public static final String LOGIN = "IvanIvanov";
  public static final String PASSWORD = "Qwerty1234";
}
```

Database

```
def getSecret(secretKey: String): String =
  readOnly {implicit session ⇒
    sql"select value from secrets where key = $secretKey"
      .map(_.string( columnLabel = "value")).single().apply().getOrElse("")
  }
```

- Pros
  - Can share configuration multiple apps
  - Easy to manage remotely via any DB tool
  - Can be cashed inside app, after loading
  - Hard to lose
- Cons
  - Performance drawbacks if often requested
  - Without tools hard to manage for administrators
  - You need another configuration method to connect to DB in the first place

Properties file

- Pros
  - Easy to manage with notepad
  - Easy to understand format
  - Can be cashed inside app, after loading
- Cons
  - Can’t share configuration multiple apps
  - Troubles with non UTF characters
  - No native support in other languages
  - Can be easily leaked
  - Depends on Java class path

Writing Properties

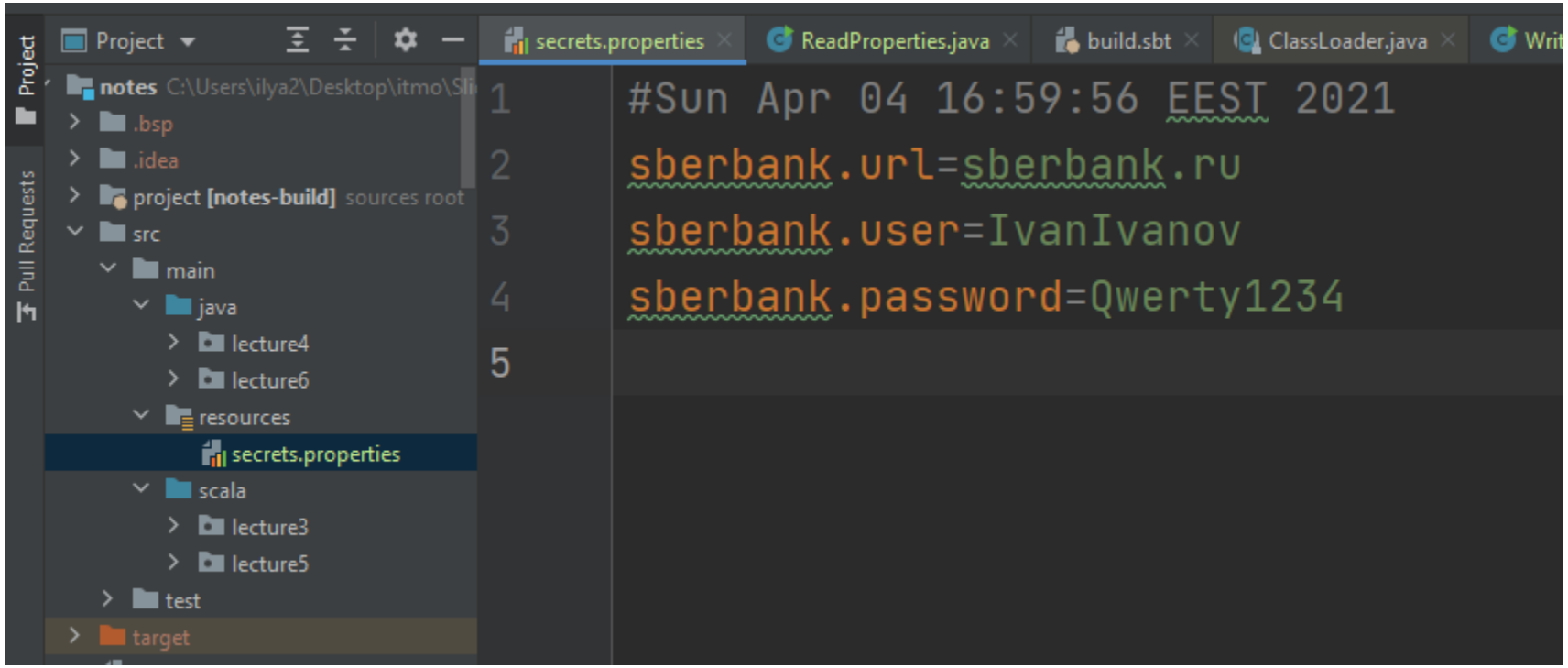
```
public class WriteProperties {
  public static void main(String[] args) {
    try (OutputStream output = new FileOutputStream( name: "./src/main/resources/secrets.properties")) {
      Properties properties = new Properties();

      properties.setProperty("sberbank.url", "sberbank.ru");
      properties.setProperty("sberbank.user", "IvanIvanov");
      properties.setProperty("sberbank.password", "Qwerty1234");

      properties.store(output,  comments: null);

      System.out.println(properties);
    } catch (IOException io) {
      io.printStackTrace();
    }
  }
}
```

Result


Project
Pull Requests
Project
notes C:\Users\ilya2\Desktop\itmo\Sli
.bsp
.idea
project [notes-build] sources root
src
main
java
lecture4
lecture6
resources
secrets.properties
scala
lecture3
lecture5
test
target
secrets.properties
ReadProperties.java
build.sbt
ClassLoader.java
Writ
1 #Sun Apr 04 16:59:56 EEST 2021
2 sberbank.url=sberbank.ru
3 sberbank.user=IvanIvanov
4 sberbank.password=Qwerty1234
5

Reading Properties

```java
public class ReadProperties {
  public static void main(String[] args) {
    try (InputStream input = ReadProperties.class.getClassLoader().getResourceAsStream( name: "secrets.properties")) {
      Properties properties = new Properties();
      if (input == null) {
        System.out.println("Sorry, unable to find secrets.properties");
        return;
      }
      properties.load(input);

      System.out.println(properties.getProperty("sberbank.url"));
      System.out.println(properties.getProperty("sberbank.username"));
      System.out.println(properties.getProperty("sberbank.password"));
    } catch (IOException exception) {
      exception.printStackTrace();
    }
  }
}
```

Environment variables

- Pros
  - Easy/Fast to obtain
  - Super hard to leak
  - Compatible with almost any deployment
- Cons
  - Basically unmanageable
  - Easy to lose

Getting environment variables

```java
public class EnvVariables {
  public static void main(String[] args) {
    Map<String, String> environment = System.getenv();
    System.out.println(environment.get("sberbank.url"));
    System.out.println(environment.get("sberbank.username"));
    System.out.println(environment.get("sberbank.password"));
  }
}
```

Environment + Properties file

```
public class ReadPropertiesEnv {
  public static void main(String[] args) {
    try (InputStream input = new FileInputStream(System.getenv( name: "SECRETS_LOCATION"))) {
      Properties properties = new Properties();
      properties.load(input);

      System.out.println(properties.getProperty("sberbank.url"));
      System.out.println(properties.getProperty("sberbank.username"));
      System.out.println(properties.getProperty("sberbank.password"));
    } catch (IOException exception) {
      exception.printStackTrace();
    }
  }
}
```

Representational state transfer

- Client-server
- Stateless
- HTTP
  - Endpoint – Root, Path, Query
  - Method
    - GET – read
    - POST – create
    - PUT/PATCH – create and/or update
    - DELETE – delete
  - Headers – Mainly authentication and body type, property-value pairs
  - Body

GitHub API create repo

2021

Create a repository for the authenticated user

Creates a new repository for the authenticated user.

OAuth scope requirements

When using OAuth, authorizations must include:

- `public_repo` scope or `repo` scope to create a public repository. Note: For GitHub AE, use `repo` scope to create an internal repository.
- `repo` scope to create a private repository.

POST /user/repos

Parameters

Name	Type	In	Description
`accept`	string	header	Setting to `application/vnd.github.v3+json` is recommended. See preview notices
`name`	string	body	**Required.** The name of the repository.
`description`	string	body	A short description of the repository.
`homepage`	string	body	A URL with more information about the repository.
`private`	boolean	body	Whether the repository is private.
`has_issues`	boolean	body	Whether issues are enabled. Default: `true`
`has_projects`	boolean	body	Whether projects are enabled. Default: `true`

GitHub API create repo

```
try {
  GitHub github = new GitHubBuilder().withOAuthToken(token).build();
  GHRepository repo = github
      .createRepository( name: "test-repo")
      .description("this is test repo")
      .gitignoreTemplate("Java")
      .homepage("test.test")
      .create();
} catch (Exception e) {
  e.printStackTrace();
}
```

List repositories

List repositories for a user

Lists public repositories for the specified user. Note: For GitHub AE, this endpoint will list internal repositories for the specified user.

GET /users/{username}/repos

```
private static void listMyRepos(GitHub github) throws IOException {
  GHUser me = github.getUser( login: "IlyaHalsky");
  PagedIterable<GHRepository> repos = me.listRepositories();
  for (GHRepository repo : repos) {
    System.out.println(repo.getName());
  }
}
```

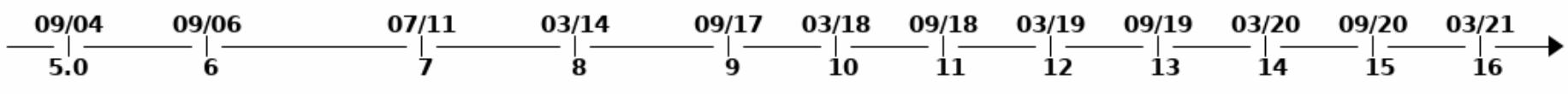
Delete Commit comments

Delete a commit comment

DELETE /repos/{owner}/{repo}/comments/{comment_id}

```
private static void deleteComments(GitHub github) throws IOException {
  GHUser owner = github.getUser( login: "IlyaHalsky");
  GHRepository repo = owner.getRepository( name: "scala-3-test");
  PagedIterable<GHCommitComment> comments = repo.listCommitComments( commitSha: "19e9e41ef1f0bdbb
  for (GHCommitComment comment: comments) {
    comment.delete();
  }
}
```

Java 8+



- Java 8 – Lambdas, Collections & Streams
- Java 9 – Collections, Streams, Optionals, Interfaces, Jshell
- Java 10 – Local-Variable Type Inference: var-keyword
- Java 11 – Strings & Files, Run Source Files, HTTP client
- Java 12 – Mainly background changes, Unicode 11
- Java 13 – Unicode 12.1
- Java 14 – Switch Expression
- Java 15 – Text-Blocks / Multiline Strings, ZGC
- Java 16 - Unix-Domain Socket Channels, Records & Pattern Matching
- Java 17 – WIP: Java 17 is renamed to Scala 2.13

Java 8

2021

```java
public class Java8 {
  public static void main(String[] args) {
    lambdas();
    anonymousClasses();
    streams();
  }

  private static void lambdas() {
    Function<Integer, Integer> test = x -> x + 2;
    System.out.println(test.apply( t: 3));
  }

  private static void anonymousClasses() {
    Runnable runnable = new Runnable(){
      @Override
      public void run(){
        System.out.println("Hello world !");
      }
    };
    Runnable runnableShorter = () -> System.out.println("Hello world two!");
    runnable.run();
    runnableShorter.run();
  }

  private static void streams() {
    List<String> list = Arrays.asList("franz", "ferdinand", "fiel", "vom", "pferd");
    list.stream()
        .filter(name -> name.startsWith("f"))
        .map(String::toUpperCase)
        .sorted()
        .forEach(System.out::println);
  }
}
```

Java 9

2021

```java
public class Java9 {
  public static void main(String[] args) {
    collections();
    streams();
    optional();
  }

  private static void optional() {
    Optional<Integer> optional = Optional.ofNullable(null);
    optional.ifPresentOrElse(Integer::toBinaryString, () -> System.out.println("It's empty"));
  }

  private static void streams() {
    Stream<String> stream = Stream.iterate( seed: "", s -> s + "s")
        .takeWhile(s -> s.length() < 10);
  }

  private static void collections() {
    List<String> list = List.of("one", "two", "three");
    Set<String> set = Set.of("one", "two", "three");
    Map<String, String> map = Map.of( k1: "foo",  v1: "one",  k2: "bar",  v2: "two");
  }

  public interface MyInterface {

    private static void myPrivateMethod(){
      System.out.println("Yay, I am private!");
    }
  }
}
```

Java 10

2021

```java
public class Java10 {
  public static void main(String[] args) {
    String myName = "Marco";
    System.out.println(myName);

    var myAnswer = "Polo";
    System.out.println(myAnswer);
  }
}
```

Java 11
Current LTS

2021

```java
public class Java11 {
  public static void main(String[] args) {
    strings();
    try {
      files();
    } catch (IOException exception) {
      exception.printStackTrace();
    }
    lambdaVar();
  }

  private static void lambdaVar() {
    BinaryOperator<String> test = (var s1, var s2) -> s1 + s2;
    System.out.println("test\ntest2\ntest3".lines().reduce( identity: "", (var s1, var s2) -> s1 + s2));
    System.out.println("test\ntest2\ntest3".lines().reduce( identity: "", test));
  }

  private static void files() throws IOException {
    Path path = Files.writeString(Files.createTempFile( prefix: "helloworld",  suffix: ".txt"),  csq: "Hi, my name is!");
    String s = Files.readString(path);
    System.out.println(s);
  }

  private static void strings() {
    var blank :boolean  = "Marco".isBlank();
    System.out.println(blank);
    var lines :Stream<String>  = "Mar\nco".lines();
    System.out.println(lines);
    var strip :String  = "Marco  ".strip();
    System.out.println(strip);
  }
}
```

Java 14

2021

```java
public class Java14 {
  public static void main(String[] args) {
    switchExpression();
  }

  enum Person {
    Mozart, Picasso, Goethe, Dostoevsky, Prokofiev, Dali
  }

  private static void switchExpression() {
    print(Person.Mozart);
    print(Person.Dali);
    print(Person.Dostoevsky);
  }

  static void print(Person person) {
    String title = switch (person) {
      case Dali, Picasso -> "painter";
      case Mozart, Prokofiev -> "composer";
      case Goethe, Dostoevsky -> "writer";
    };
    System.out.printf("%s was a %s%n", person, title);
  }
}
```

Java 15

```java
public class Java15 {
  public static void main(String[] args) {
    String text = """
        Lorem ipsum dolor sit amet, consectetur adipiscing \
        elit, sed do eiusmod tempor incididunt ut labore \
        et dolore magna aliqua.\
        """;
  }
}
```

Java 16

2021

```java
public class Java16 {

  public static void main(String[] args) {
    records();
    patternMatching();
    patternMatchingInIf();
  }

  record Point(int x, int y) {}
  record Rectangle(Point a, Point b) { int length() { return 10; }}
  private static void records() {
    var point = new Point( x: 1,  y: 2);
    System.out.println(point.x);
    System.out.println(point.y);
  }

  private static void patternMatching() {
    var obj = new String( original: "hello");
    if (obj instanceof String s) {
      System.out.println(s.contains("hello"));
    }
  }

  private static void patternMatchingInIf() {
    var a = new Rectangle(new Point( x: 1, y: 2), new Point( x: 2, y: 3));
    if (a instanceof Rectangle s && s.length() > 5) {
      System.out.println("Long boy");
    } else {
      System.out.println("Short boy");
    }
  }
}
```

Useful links

- GitHub API docs - https://docs.github.com/en/rest
- GitHub API for Java - https://github-api.kohsuke.org/
- Java 8-16 - https://www.marcobehler.com/guides/a-guide-to-java-versions-and-features#_java_features_8_16
- Java 8 - https://www.baeldung.com/java-8-new-features
- Java 9 - https://www.baeldung.com/new-java-9
- Java 10 - https://www.baeldung.com/java-10-overview
- Java 11 - https://www.baeldung.com/java-11-new-features
- Java 12 - https://www.baeldung.com/java-12-new-features
- Java 13 - https://www.baeldung.com/java-13-new-features
- Java 14 - https://www.baeldung.com/java-14-new-features
- Java 15 - https://www.baeldung.com/java-15-new
- Java 16 - https://www.azul.com/67-new-features-in-jdk-16/
- ZGC - https://wiki.openjdk.java.net/display/zgc/Main