

Lombok

Also look: Google AutoValue

Azat Satklichov

azats@seznam.cz,

<http://sahet.net/htm/java.html>,

<https://github.com/azatsatklichov/java-and-ts-tests/tree/master/javatesting>



Lombok

<https://projectlombok.org/>



The word Lombok (island in Indonesia) is a word that comes from the local Sasak language. Translated into Indonesian it means lurus or straight. Lombok is also a lesser-used word for chili in Bahasa Indonesian, which has led many people to believe the island is named for its **spicy cuisine**.

v1.18.16 (October 15th, 2020)

.. v1.18.14 (broken), v1.18.12() – Java 13,14 (yield)

v1.18.10 (September 10th, 2019)

Lombok, is a library used to reduce boilerplate code (simplifying data objects) for model/data objects. e.g., can generate getters and setters, toString, hashCode,

- @Slf4j|@Log4j|@Log, @NonNull, @Synchronized, @Getter, @Setter, @ToString, @EqualsAndHashCode - provide logger, puts null-check (NPEs), thread-safety, getters, setters, toString, equals and hashCode
`javax.annotation.Nullable` is part of JSR-305, **dormant** since 2012. Used by *Findbugs*, .. *SonarQube*
- Guava -> forces us to require jsr305 automatic module,
- @ToString(exclude = {"events"}), or on field @ToString.Exclude ?? StackOverflow.. (biDir)
- @EqualsAndHashCode(of = {"authToken"}). Also can be **excluded**

All fields marked as **static**, **transient** will not be considered for **hashCode** and **equals**

- @Getter(**AccessLevel.PRIVATE**) ... `Waiter()`; `DatabusBrokerTest`
 - Lazy Getter (performance: cache it to allow in-memory reads or retrieve when its needed)

```
@Getter(lazy = true) private final Map<String, Long> transactions =
getTransactions();
```

- @Accessors(fluent = .., chain = .., prefix) e.g. `DatabusBrokerTest`, `DatabusConfigProvider`
e.g. `private BigDecimal bdBalance;` → `@Accessors(prefix = {"bd"})` ->
`obj.getBalance()`
 - Also project-wide: **Lombok.config**

- @NoArgsConstructor, @RequiredArgsConstructor (for the final and @NonNull fields), @AllArgsConstructor - DataEvent, UnknownEventSubscriber(X)

- - for all e.g. @AllArgsConstructor(staticName = "of"), - allows creation of static factory method

@Data - get all this for free: A shortcut for @ToString, @EqualsAndHashCode, @Getter on all fields, @Setter on all non-final fields, and @RequiredArgsConstructor!

e.g. UnknownEvent, CobolDocumentModel

- staticConstructor = "of" constructor and a public static factory method:

```
@Data(staticConstructor = "of")
public class Employee {

    private String name;
    private int salary;
}
```

```
private Employee() {
}

public static Employee of() {
    return new Employee();
}

// Other methods...
```

- DTO - @Value (thread safe), immutable entity (final class with imm. members, @Value is immutable variant of @Data): No setter by Default, constructor arguments (except final fields that are initialized in the field declaration) is also generated. @Value is shorthand for: final @ToString @EqualsAndHashCode @AllArgsConstructor @FieldDefaults(makeFinal = true, level = AccessLevel.PRIVATE) @Getter. Also, any explicit constructor, no matter the arguments list, implies lombok will not generate a constructor

e.g. Locality, ElementItem

- @Value(staticConstructor = "of"), to make some fields @NonFinal

- **@Builder** - build immutable data objects with their simple, fluent syntax. If needed a builder for specific fields, we should create a constructor/method with only those fields.
- **@Singular** - the builder doesn't generate a *setter* method. Instead, it generates two *adder* methods.

e.g. `SyntaxError`, `AnalysisFinishedEvent`

`.rules(Arrays.asList("rule1","rule2"))` → `.rule("rule1").rule("rule2")`.

- Builder with Default Value

1. via `toBuilder()`

2. **@Builder.Default** - e.g. `AnalysisFinishedEvent`

lombok.singular.useGuava to true, Lombok uses Guava's immutable builders and types.

CheckedExceptions and Ensure Your Resources Are Released

@SneakyThrows, **@Cleanup**

e.g. Messages

Experimental: Automate Objects Composition - “favor composition inheritance”, (*Traits or Mixins*)

```
public interface HasContactInformation {

    String getFirstName();
    void setFirstName(String firstName);

    String getFullName();

    String getLastName();
    void setLastName(String lastName);

    String getPhoneNr();
    void setPhoneNr(String phoneNr);

}

@Data
public class ContactInformationSupport implements HasContactInformation {

    private String firstName;
    private String lastName;
    private String phoneNr;

    @Override
    public String getFullName() {
        return getFirstName() + " " + getLastName();
    }

}

public class User implements HasContactInformation {

    // Whichever other User-specific attributes

    @Delegate(types = {HasContactInformation.class})
    private final ContactInformationSupport contactInformation =
        new ContactInformationSupport();

    // User itself will implement all contact information by delegation

}
```

Experimental Lombok - <https://projectlombok.org/features/experimental/all>

@UtilityClass

[@Accessors](#)

@Helper - you can declare classes inside methods

@Delegate

Val: val example = **new** ArrayList<String>();,

CobolLineIndicatorProcessorImplTest

Var(promoted): **var** (mutable) like [val](#), but *not* marked as **final**

@Tolerate - lombok pretend it doesn't exist, i.e., to generate a method which would otherwise be skipped due to possible conflicts. Similar concept: **Hijack in ClearCase (or ignore)**

@Jacksonized - v1.18.14: add-on annotation for [@Builder](#), [@SuperBuilder](#),

auto- configures builder to be used by [Jackson](#)'s deserialization

[Promoted](#): @Value, @Builder, @Wither: renamed to @With, and promoted Immutable 'setters' - methods that create a clone but with one changed field.

```
public class HelperExample {
    int someMethod(int arg1) {
        int localVar = 5;

        @Helper class Helpers {
            int helperMethod(int arg) {
                return arg + localVar;
            }
        }

        return helperMethod(10);
    }
}
```

```
@Jacksonized @Builder
@JsonIgnoreProperties(ignoreUnknown = true)
public class JacksonExample {
    private List<Foo> foos;
}
```


- Delombok: `java -jar lombok.jar delombok src -d src-delomboked`
- Or and Ant-task:

Delombok tries to preserve your code as much as it can, but comments may move around a little bit, especially comments that are in the middle of a syntax node.

- `@Builder` - build immutable data objects with their simple, fluent syntax
`@SuperBuilder` - <https://www.baeldung.com/lombok-builder-inheritance>
- `@Setter` → `@With`: The next best alternative to a setter for an immutable property is to construct a clone of the object, but with a new value for this one field.
- **if the superclass doesn't have a no-args constructor, Lombok can't generate any constructor in the subclass**



THANK YOU