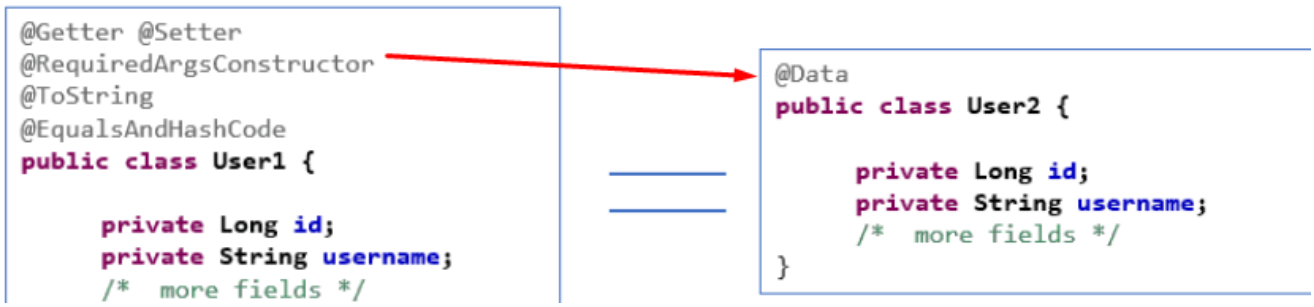
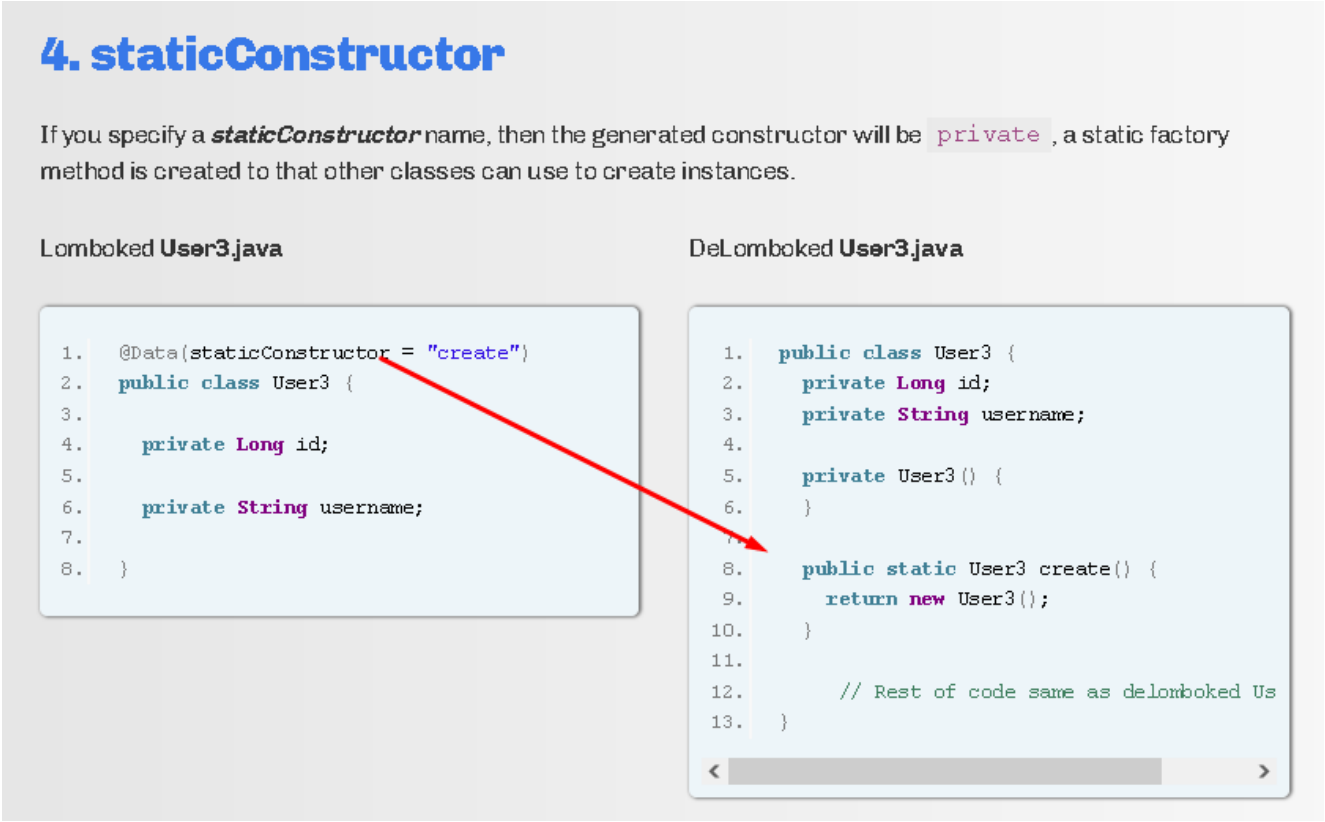
Lombok

[crayon-66ecefe83f072767107652/]



Ref: javabydeveloper.com (All the content!)

[crayon-66ecefe83f07a276072258/]
[crayon-66ecefe83f07d993690737/]
[crayon-66ecefe83f07f684047429/]



[crayon-66ecefe83f082714619322/]

[crayon-66ecefe83f084118888305/]

---

4. staticConstructor

If you specify a ***staticConstructor*** name, then the generated constructor will be `private`, a static factory method is created to that other classes can use to create instances.

Lomboked **User3.java**

```
1.  @Data(staticConstructor = "create")
2.  public class User3 {
3.
4.    private Long id;
5.
6.    private String username;
7.
8.  }
```

DeLomboked **User3.java**

```
1.  public class User3 {
2.    private Long id;
3.    private String username;
4.
5.    private User3() {
6.    }
7.
8.    public static User3 create() {
9.      return new User3();
10.   }
11.
12.     // Rest of code same as delomboked Us
13. }
```

Sometimes you may want to define annotations on top of constructor, for example when you are working with Spring framework or some other third part java libraries, you may need to declare annotations on top of constructor. `onConstructor` attribute of `@AllArgsConstructor` allows us to put annotations on generated all-args constructor.

1. Up to JDK7: `@AllArgsConstructor(onConstructor=@__({@AnnotationsGoHere}))`
2. From JDK8: `@AllArgsConstructor(onConstructor_={@AnnotationsGohere})` // note the underscore after `onConstructor`.

Lomboked ***AllArgsDemo6.java***

```
1.  @AllArgsConstructor(
2.      onConstructor_=
3.      @ConstructorProperties({"id", "usernam
4.  public class AllArgsDemo6 {
5.
6.    private Long id;
7.
8.    private String username;
9.
10. }
```

DeLomboked ***AllArgsDemo6.java***

```
1.  public class AllArgsDemo6 {
2.    private Long id;
3.    private String username;
4.
5.    @ConstructorProperties({"id", "username"
6.    public AllArgsDemo6(final Long id, final
7.      this.id = id;
8.      this.username = username;
9.    }
10. }
```

If we would like to create instance using a `static` factory method, `staticName` attribute of `@AllArgsConstructor` allows us to generates a private all-args constructors and an additional `static` factory method that wraps around the `private` constructor is generated. Let's have a look into following example.

Lomboked ***AllArgsDemo5.java***

```
@AllArgsConstructor(staticName = "getInstance")
public class AllArgsDemo5 {

  private Long id;

  private String username;

}
```

DeLomboked ***AllArgsDemo5.java***

```
public class AllArgsDemo5 {
  private Long id;
  private String username;

  private AllArgsDemo5(final Long id, final Str
    this.id = id;
    this.username = username;
  }

  public static AllArgsDemo5 getInstance(final
    return new AllArgsDemo5(id, username);
  }
}
```

Lombok generates a public all-args constructor by default for the ***`@AllArgsConstructor`***. To generate `private` all-args constructor define `@AllArgsConstructor(access = AccessLevel.PRIVATE)`. `access` attribute of ***`@AllArgsConstructor`*** allows you to change the access modifier of the generated constructor.

Lomboked ***AllArgsDemo4.java***

```
@AllArgsConstructor(access = AccessLevel.PRIVATE
public class AllArgsDemo4 {

  private Long id;

  private String username;

}
```

DeLomboked ***AllArgsDemo4.java***

```
public class AllArgsDemo4 {
  private Long id;
  private String username;

  private AllArgsDemo4(final Long id, final Strin
    this.id = id;
    this.username = username;
  }
}
```

1. Lombok never generates constructor argument for the `static` fields for `@AllArgsConstructor`.
2. For `@AllArgsConstructor` Lombok never generates constructor argument for the `final` fields if they are initialized with value, otherwise an argument will be generated.

Lomboked ***AllArgsDemo3.java***

```
1.  @AllArgsConstructor
2.  public class AllArgsDemo3 {
3.
4.    private Long id;
5.
6.    private static boolean defaultStatus;
7.
8.    private final double minSalary = 10000.00;
9.
0.    private final int defaultRole;
1.  }
```

DeLomboked ***AllArgsDemo3.java***

```
blic class AllArgsDemo3 {
 private Long id;
 private static boolean defaultStatus;
 private final double minSalary = 10000.0;
 private final int defaultRole;

 public AllArgsDemo3(final Long id, final int def
   this.id = id;
   this.defaultRole = defaultRole;
 }
```

The fields marked with `@NonNull` in your class result in null check on those parameter within generated all-args constructor.

Lomboked ***AllArgsDemo2.java***

```
1.  @AllArgsConstructor
2.  public class AllArgsDemo2 {
3.
4.    private Long id;
5.
6.    @NonNull
7.    private String username;
8.  }
```

DeLomboked ***AllArgsDemo2.java***

```
1.  public class AllArgsDemo2 {
2.    private Long id;
3.    @NonNull
4.    private String username;
5.
6.    public AllArgsDemo2(final Long id, @NonN
7.      if (username == null) {
8.        throw new NullPointerException("user
9.      }
10.     this.id = id;
11.     this.username = username;
12.   }
13. }
```

```
1.  @AllArgsConstructor
2.  public class AllArgsDemo1 {
3.
4.    private Long id;
5.
6.    private String username;
7.  }
```

```
1.  public class AllArgsDemo1 {
2.    private Long id;
3.    private String username;
4.
5.    public AllArgsDemo1(final Long id, final
6.      this.id = id;
7.      this.username = username;
8.    }
9.  }
```

Delombok

[crayon-66ecefe83f086544856463/]
—

—

—

—