|  |  |
| --- | --- |
| @NoArgsConstructor | Umiejscowienie: Nad klasą   * Zapewnia konstruktor bez parametrów * Jeśli pola klasy mają wartość final należy użyć parametru adnotacji force=true, aby pola były inicjalizowane z wartościami 0 / false / null * Dobre rozwiązanie dla Encji Hibernate   @NoArgsConstructor **public static class**NoArgsExample {   @NonNull **private**String field; }  <https://projectlombok.org/features/constructor> |
| @AllArgsConstruktor | Umiejscowienie: Nad klasą   * Zapewnia konstruktor z 1 parametrem dla każdego pola w klasie * Pola oznaczone adnotacją @NonNull będą sprawdzane pod względem wartości null * Dobre rozwiązanie dla Encji Hibernate   @AllArgsConstructor(access = AccessLevel.PROTECTED) **public static class**ConstructorExample<T>{   **private int**x, y;   @NonNull  **private**T description;  }  <https://projectlombok.org/features/constructor> |
| @NonNull | Umiejscowienie: przed parametrem lub polem   * Zapewnia że pole nie będzie miało wartości null   **public class**NonNullExample **extends**Something {  @NonNull   **private**String name;      **public**NonNullExample(@NonNull Person person) {     **super**("Hello");     **this**.name = person.getName();   }  }  <https://projectlombok.org/features/NonNull> |
| @RequiredArgsCunstructor | Umiejscowienie: Nad klasą   * Zapewnia konstruktor z 1 parametrem dla każdego pola w klasie, który wymaga specjalnego traktowania. * Dostarczy parametry dla pól final i NonNull * Pola oznaczone adnotacją @NonNull będą sprawdzane pod względem wartości null * Dobre rozwiązanie dla Controllera Hibernate’a   @RequiredArgsConstructor(staticName = "of") **public static class**ConstructorExample<T>{   **private int**x, y;   @NonNull  **private**T description;  }  <https://projectlombok.org/features/constructor> |
| @EqualsAndHashCode | Umiejscowienie: Nad klasą  - Generuje metody HashCode i Equals dla wszystkich pól - Modyfikacje:  Ustawienie dla wybranych pól:  @EqualsAndHashCode(of={a, b, ...})  lub  @EqualsAndHashCode(onlyExplicitlyIncluded = true)  Nad polem:  @EqualsAndHashCode.Include - pole ujęte  @EqualsAndHashCode.Exclude - pole wykluczone  @EqualsAndHashCode **public class**EqualsAndHashCodeExample {   **private transient int**transientVar = 10;  @Getter  **private**String name;  **private double**score;   @EqualsAndHashCode.Exclude  **private**Shape shape = **new**Square(5, 10);   **private**String[] tags;   @EqualsAndHashCode.Exclude  **private int**id; }  <https://projectlombok.org/features/EqualsAndHashCode> |
| @Data | Umiejscowienie: Nad klasą   * Zapewnia konstruktor dla pól NonNull / final * Zapewnia publiczne gettery i settery dla pól NonNull / final * Zapewnia Equals i hashCode dla wszystkich pół   @Data  **public class**DataExample {   **private final**String name;   @Setter(AccessLevel.PACKAGE)  **private int**age;   **private double**score;   **private**String[] tags;      @ToString(includeFieldNames=**true**)   @Data(staticConstructor="of")   **public static class**Exercise<T> {     **private final**String name;     **private final**T value;   } }  <https://projectlombok.org/features/Data> |
| @Getter i @Setter | Umiejscowienie: Nad nazwą lub polem klasy  Przed nazwą klasy:   * Zapewnia publiczne gettery lub settery dla wszystkich pól klasy   Przed polem klasy:   * Zapewnia publiczny getter i setter   Ważne modyfikatory:  **AccessLevel** pozwala zmienić poziom akcesora na:  public | protected | package | private  **public class**GetterSetterExample {     @Getter @Setter  **private int**age = 10;     @Setter(AccessLevel.PROTECTED)  **private**String name;    @Override  **public**String toString() {     **return**String.format("%s (age: %d)", name, age);   } }  <https://projectlombok.org/features/GetterSetter> |