Milou fait un régime
Guava - Lombok

Lyon JUG
Java User Group



Lyon JUG

OBJET DIRECT

zenika
ARCHITECTURE INFORMATIQUE

SPONSORS

Atos
Worldline

@thierryleriche

Thierry Leriche-Dessirier

- Consultant JEE freelance
- Professeur de Génie Logiciel à l'ESIEA
- Rédacteur pour Programmez
- Rédacteur pour Developpez.com

icauda.com

Developpez.com
Club des développeurs

- Cours
- Forums
- FAQ
- News
- Interviews
- Critiques
- Articles / tutoriels
- Agendas
- Magazine

Developpez.com
Club des développeurs

13 000 000 pages vues par mois
5 500 000 visites par mois
2 500 000 visites uniques par mois
5 000 messages forum par jour

Agenda

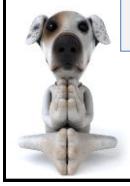
- Milou est trop lourd (intro)
- Lombok (et Lombok-pg) en action
- Tour d'horizon de Guava

Milou est trop lourd (et doit faire un régime)

I am Milou

DOG

```
public class Dog {
  private Integer       id ;
  private String        name ;
  private String        fullName ;
  private SexeEnum      sex ;
  private Date          birthday ;
  private String        race ;
  private Boolean       lof ;
  private Double        weight ;
  private Double        size ;
  private List<String>  colors ;
  ...
```



- Constructeurs
- Getters / setters
- toString ()
- equals ()
- hashCode ()
- compareTo ()

- id
- name
- fullName
- sex
- birthday
- race
- lof
- weight
- size
- Colors

- id
- name
- fullName
- sex
- birthday
- race
- lof
- weight
- size
- Colors

eclipse

clic clic
démo

- Constructeurs
- Getters / setters
- toString ()
- equals ()
- hashCode ()
- compareTo ()

```
public class Dog
       implements Comparable<Dog> {

  ...

  @Override
  public int compareTo(Dog other) {
    int result = 0;

    result = name.compareTo(other.name);
    if (result != 0) {
      return result;
    }

    ...
  }
```

Code très limite… NPE ?…

- id
- name
- fullName
- sex
- birthday
- race
- lof
- weight
- size
- Colors

- Constructeurs
- Getters / setters
- toString ()
- equals ()
- hashCode ()
- compareTo ()

10 attributs

← 210 lignes

Java

10 attributs

210 lignes de code

Va chercher Guava ☺

- toString ()
- equals ()
- hashCode ()
- compareTo ()

Commons → FAQ 1-2

Base

Objects

Java 5

```
public String toString() {
   return "Dog [id=" + id
     + ", name=" + name
     + ", fullName=" + fullName
     + ", sex=" + sex
     + ", birthday=" + birthday
     + ", race=" + race
     + ", lof=" + lof
     + ", weight=" + weight
     + ", size=" + size
     + ", colors=" + colors + "]";
}
```

Simple, efficace et bien pourri ☹

- toString ()

Java 5

```
public String toString() {
   StringBuilder sb = new StringBuilder();
   sb.append(Dog.class.getSimpleName())
     .append("[id=").append(id)
     .append(", name=").append(name)
     .append(", fullName=").append(fullName)
     ...
     .append(", colors=").append(colors);
   return sb.toString();
}
```

Mieux mais sans plus ☹

- toString ()

Guava

```
public String toString() {
   return Objects.toStringHelper(this)
            .add("id", id)
            .add("name", name)
            .add("fullName", fullName)
            ...
            .add("colors", colors)
            .toString();
}
```

Builder

- toString ()

Java 5

birthday, fullname, name, race et sex

```
public boolean equals(Object obj) {
  if (this == obj) return true;
  if (obj == null) return false;
  if (getClass() != obj.getClass()) return false;

  Dog other = (Dog) obj;
  if (birthday == null) {
    if (other.birthday != null) return false;
  } else if (!birthday.equals(other.birthday)) return false;
  if (fullName == null) {
    if (other.fullName != null) return false;
  ...
  } else if (!race.equals(other.race)) return false;
  if (sex != other.sex) return false;

  return true;
}
```

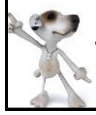
Carrément illisible ☹

- equals ()

Guava

birthday, fullname, name, race et sex

```
public boolean equals(Object obj) {
  if (!(obj instanceof Dog)) return false;

  Dog other = (Dog) obj;

  return Objects.equal(birthday, other.birthday)
         && Objects.equal(fullname, other.fullname)
         && Objects.equal(name, other.name)
         && Objects.equal(race, other.race)
         && sex == other.sex;
}
```



- equals ()

Java 5

birthday, fullname, name, race et sex

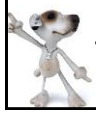
```
public int hashCode() {
  final int prime = 31;
  int result = 1;

  result = prime * result
      + ((birthday == null) ? 0 : birthday.hashCode());
  result = prime * result
      + ((fullName == null) ? 0 : fullName.hashCode());
  result = prime * result
      + ((name == null) ? 0 : name.hashCode());
  result = prime * result
      + ((race == null) ? 0 : race.hashCode());
  result = prime * result
      + ((sex == null) ? 0 : sex.hashCode());

  return result;
}
```

- hashCode ()

Guava

birthday, fullname, name, race et sex

```
public int hashCode() {
  return Objects.hashCode(birthday, fullName,
                          name, race, sex);
}
```



- hasCode()

Java 5

name, fullname, birthday, weight, size, race et sex

```
public int compareTo(Dog other) {
  int result = 0;

  result = name.compareTo(other.name);
  if (result != 0) {
    return result;
  }
  result = fullname.compareTo(other.fullname);
  if (result != 0) {
    return result;
  }

  ...
}
```

- compareTo ()

```
name.compareTo(other.name)
```

Vs

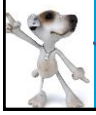
```
other.name.compareTo(name)
```

- compareTo ()

Guava

name, fullname, birthday, weight, size, race et sex

```
public int compareTo(Dog other) {
  return ComparisonChain.start()
            .compare(name, other.name)
            .compare(fullName, other.fullName)
            .compare(birthday, other.birthday)
            .compare(weight, other.weight)
            .compare(size, other.size)
            .compare(race, other.race)
            .compare(sex, other.sex)
            .result();
}
```



- compareTo ()

Lombok

en action

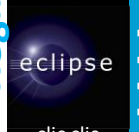
Régime

Annotations de base

Régime

- @NoArgsConstructor / @RequiredArgsConstructor / @AllArgsConstructor → constructeurs
- @Getter / @Setter
- @ToString
- @EqualsAndHashCode
- @Data → @Getter + @Setter + @ToString + @EqualsAndHashCode + @RequiredArgsConstructor

Régime



clic clic
démo

- @NoArgsConstructor / @RequiredArgsConstructor / @AllArgsConstructor
- @Getter / @Setter
- @ToString
- @EqualsAndHashCode
- @Data

10 attributs

Java 

210 lignes de code

Lombok 

12 lignes + 2-3 @ + compareTo()

Factory

```
@RequiredArgsConstructor(staticName = "of")
public class Dog {
  private Integer id;
  @NonNull private String name;
  private String fullName;
  @NonNull private SexeEnum sex;
  private Date birthday;
  ...
```

name, sex

```
Dog dog = Dog.of("Milou", MALE);
```

Factory

```
@RequiredArgsConstructor(staticName = "of")
public class Dog {
  private Integer id;
  @NonNull private String name;
  private String fullName;
  @NonNull private SexeEnum sex;
  private Date birthday;
  ...
```

name, sex

```
Dog dog = Dog.of("Milou", MALE);
```

```
private Dog(@NonNull final String name,
            @NonNull final SexeEnum sex) {
  if (name == null) throw new NullPointerException("name");
  if (sex == null) throw new NullPointerException("sex");
  this.name = name;
  this.sex = sex;
}
public static Dog of(@NonNull final String name,
                     @NonNull final SexeEnum sex) {
  return new Dog(name, sex);
}
```

Trucs pratiques

Se simplifier la vie

Pratique

- @Cleanup
- @Synchronized
- @SneakyThrows
- @Log
- @Delegate

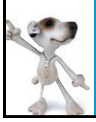
- *val*

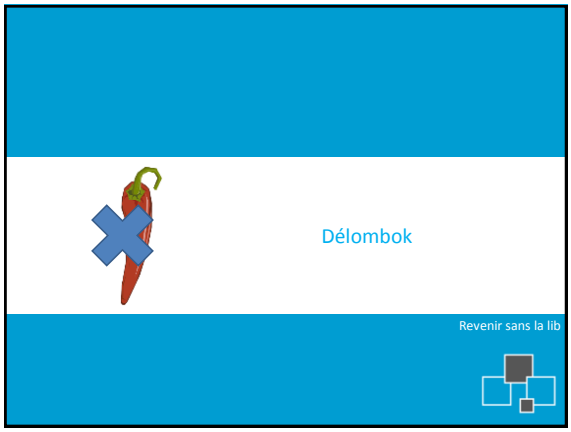
Ressources

```
public void lireJava6(String from) {
  InputStream in = null;
  try {
    in = new FileInputStream(from);
    byte[] b = new byte[10000];
    while (true) {
      int r = in.read(b);
      if (r == -1)
        break;
      ...
    }
  } catch (Exception e) {
    ...
  } finally {
    if (in != null) {
      try {
        in.close();
      } catch (Exception e) {
        ...
      }
    }
  ...
```

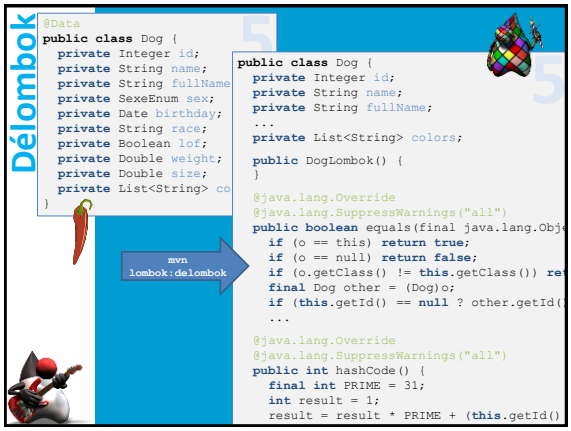
Ressources

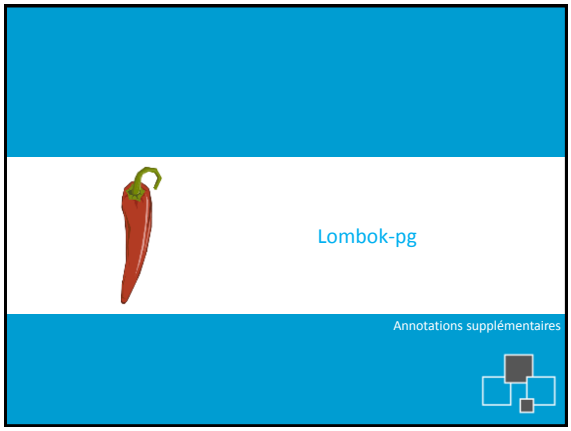
```
public void lire(String from) throws IOException {
  @Cleanup
  InputStream in = new FileInputStream(from);
  byte[] b = new byte[10000];
  while (true) {
    int r = in.read(b);
    if (r == -1)
      break;
    ...
  }
}
```

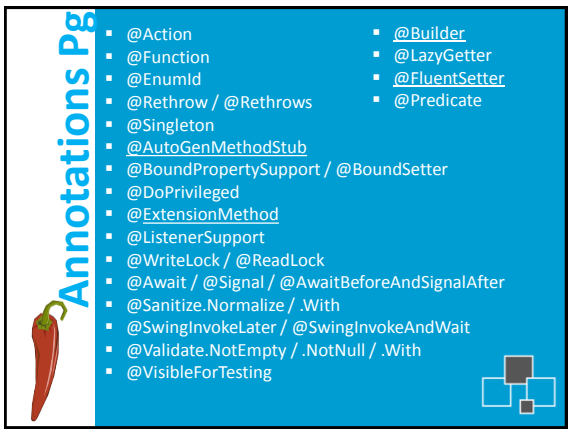


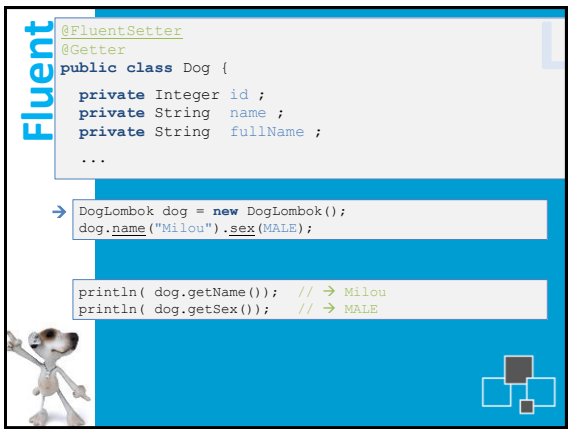

Délombok
Revenir sans la lib


Délombok
java -jar lombok.jar delombok src -d src-delomboked
XML
<properties>
<lombok.sourceDirectory>
${project.basedir}/src/main/java
</lombok.sourceDirectory>
...
<build>
<groupId>org.projectlombok</groupId>
<artifactId>maven-lombok-plugin</artifactId>
<executions>
<execution>
<phase>generate-sources</phase>
</execution>
</executions>
...
mvn lombok:delombok


Délombok
@Data
public class Dog {
private Integer id;
private String name;
private String fullName
private SexeEnum sex;
private Date birthday;
private String race;
private Boolean lof;
private Double weight;
private Double size;
private List<String> co
}
mvn
lombok:delombok
public class Dog {
private Integer id;
private String name;
private String fullName;
...
private List<String> colors;
public DogLombok() {
}
@java.lang.Override
@java.lang.SuppressWarnings("all")
public boolean equals(final java.lang.Obj
if (o == this) return true;
if (o == null) return false;
if (o.getClass() != this.getClass()) re
final Dog other = (Dog)o;
if (this.getId() == null ? other.getId(
...
@java.lang.Override
@java.lang.SuppressWarnings("all")
public int hashCode() {
final int PRIME = 31;
int result = 1;
result = result * PRIME + (this.getId()


Lombok-pg
Annotations supplémentaires


Annotations Pg
@Action
@Function
@EnumId
@Rethrow / @Rethrows
@Singleton
@AutoGenMethodStub
@BoundPropertySupport / @BoundSetter
@DoPrivileged
@ExtensionMethod
@ListenerSupport
@WriteLock / @ReadLock
@Await / @Signal / @AwaitBeforeAndSignalAfter
@Sanitize.Normalize / .With
@SwingInvokeLater / @SwingInvokeAndWait
@Validate.NotEmpty / .NotNull / .With
@VisibleForTesting
@Builder
@LazyGetter
@FluentSetter
@Predicate


Fluent
@FluentSetter
@Getter
public class Dog {
private Integer id ;
private String name ;
private String fullName ;
...
DogLombok dog = new DogLombok();
dog.name("Milou").sex(MALE);
println(dog.getName()); // → Milou
println(dog.getSex()); // → MALE

Builder

```
@Builder
@Getter
public class Dog {
  private Integer id ;
  private String  name ;
  private String  fullName ;
  ...
```

```
Dog dog = Dog.dog().name("Milou").sex(MALE).build();
```

```
println( dog.getName());   // → Milou
println( dog.getSex());    // → MALE
```

extension

```
@ExtensionMethod({ Dog.class, MyDogExtensionClass.class })
public class DogWithExtension {
  public void foo() {
    Dog milou = new Dog("Milou", 12.5, ...);
    boolean isTooFat = milou.isTooFat();
  }
}
class MyDogExtensionClass {
  public static boolean isTooFat(final Dog dog) {
    double imc = dog.getWeight() / pow(dog.getSize(), 2);
    return 25 < imc;
  }
}
```

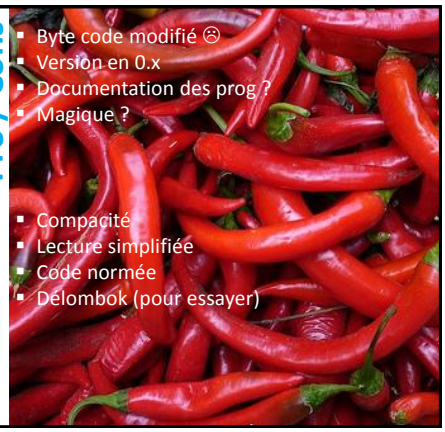
Avec paramètres → FAQ 9

→ http://blog.developpez.com/todaystip/p11165/dev/java/extensionmethod-de-lombok-pg/

Lombok or not Lombok ?

Avantages et inconvénients

Pro / Cons



- Byte code modifié ☹
- Version en 0.x
- Documentation des prog ?
- Magique ?

- Compacité
- Lecture simplifiée
- Code normée
- Délombok (pour essayer)

Guava

tour d'horizon

Factory Methods

Les choses que j'aime

< new Vs static factories >

Java 5

```
List<Integer> primeNumbers = new ArrayList<Integer>();
Set<String> colors = new TreeSet<String>();
Map<String, Integer> ages = new HashMap<String, Integer>();
```

< new Vs static factories >

Java 5

```
List<Integer> primeNumbers = new ArrayList<Integer>();
Set<String> colors = new TreeSet<String>();
Map<String, Integer> ages = new HashMap<String, Integer>();
```

dur dur ?

```
Map<String, List<String>> trucs =
    new HashMap<String, List<String>>();

Map<? extends Person, Map<String, List<String>>> trucs =
    new TreeMap<? extends Person, Map<String, List<String>>>();

Map<? extends Wrapper<String, Sex, Person>, Map<String,
List<Set<Adress<String, Integer, Country>>>>> trucs = ...
```

< new Vs static factories >

Java 5

```
List<Integer> primeNumbers = new ArrayList<Integer>();
Set<String> colors = new TreeSet<String>();
Map<String, Integer> ages = new HashMap<String, Integer>();
```

Java 7

```
List<Integer> primeNumbers = new ArrayList<>();
Set<String> colors = new TreeSet<>();
Map<String, Integer> ages = new HashMap<>();
```

Qui utilise Java 7 en prod ?

< new Vs static factories >

Java 5

```
List<Integer> primeNumbers = new ArrayList<Integer>();
Set<String> colors = new TreeSet<String>();
Map<String, Integer> ages = new HashMap<String, Integer>();
```

Java 7

```
List<Integer> primeNumbers = new ArrayList<>();
Set<String> colors = new TreeSet<>();
Map<String, Integer> ages = new HashMap<>();
```

Qui utilise Java 7 en prod ?

Guava

```
List<Integer> primeNumbers = newArrayList();
Set<String> colors = newTreeSet();
Map<String, Integer> ages = newHashMap();
```

Dès maintenant !

< new Vs static factories >

Java 5

```
List<Integer> primeNumbers = new ArrayList<Integer>();
Set<String> colors = new TreeSet<String>();
Map<String, Integer> ages = new HashMap<String, Integer>();
```

Java 7

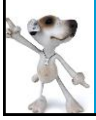
```
List<Integer> primeNumbers = new ArrayList<>();
Set<String> colors = new TreeSet<>();
Map<String, Integer> ages = new HashMap<>();
```

Qui utilise Java 7 en prod ?

Guava

```
List<Integer> primeNumbers = newArrayList();
Set<String> colors = newTreeSet();
Map<String, Integer> ages = newHashMap();
```

Dès maintenant !



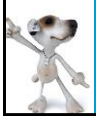
Lombok

```
var primeNumbers = new ArrayList<Integer>();
-> primeNumbers.size();
   ...
```

< new Vs static factories >

List of dogs

```
List<Dog> dogs = newArrayList(
    new Dog("Milou", 12.5, MALE, ...),
    new Dog("Rintintin", 45.0, MALE, ...),
    new Dog("Volt", 10.3, MALE, ...),
    new Dog("Lassie", 45.0, FEMALE, ...),
    new Dog("Pluto", 22.0, MALE, ...),
    new Dog("Medor", 35.6, MALE, ...));
```



[Milou,
Rintintin,
Volt,
Lassie,
Pluto,
Medor]

Immutables

Pour que ça ne change pas

immutable Quand ?

immutable Quand ? mutable

On se demande souvent si une liste doit être immutable mais c'est prendre le problème dans le mauvais sens. La plupart du temps, ce qu'il faut vraiment se demander, c'est si la liste a besoin d'être mutable.

< unmodifiables Vs immutables >

Java 5

```
Set<Integer> temp =
    new LinkedHashSet<Integer>(Arrays.asList(1, 2, 3, 5, 7));
Set<Integer> primes = Collections.unmodifiableSet(temp);
```

< unmodifiables Vs immutables >

Java 5

```
Set<Integer> temp =
    new LinkedHashSet<Integer>(Arrays.asList(1, 2, 3, 5, 7));
Set<Integer> primes = Collections.unmodifiableSet(temp);
```

Guava

```
Set<Integer> primes = ImmutableSet.of(1, 2, 3, 5, 7);
```

< unmodifiables Vs immutables >

Of

```
ImmutableSet.of(E e1)
ImmutableSet.of(E e1, E e2)
ImmutableSet.of(E e1, E e2, E e3)
ImmutableSet.of(E e1, E e2, E e3, E e4)
ImmutableSet.of(E e1, E e2, E e3, E e4, E e5)
ImmutableSet.of(E e1, E e2, E e3, E e4, E e5, E e6, E...)
```

Ne prend pas de null

Of

```
ImmutableSet.of()
```

Vide

Map et List → FAQ 4

→ http://blog.developpez.com/guava/p10589/collection/title-212/

< Copie de défense >

Java 5

```
public class Dog {
  private String name;
  ...
  private List<String> colors;

  public Dog(String name, List<String> colors) {
    this.name = name;
    ...
    this.colors = Collections.unmodifiableList(
                    new ArrayList<String>(colors));
  }
  public List<String> getColors() {
    return colors;
  }
```

< Copie de défense >

Guava

```
public class Dog {
  private String name;
  ...
  private ImmutableList<String> colors;

  public Dog(String name, List<String> colors) {
    this.name = name;
    ...
    this.colors = ImmutableList.copyOf(colors);
  }

  public ImmutableList<String> getColors() {
    return colors;
  }
```

Message clair

Collections en plus

Multimap, Bipap, Multiset

< Multi Map >

Java 5

```
Map<String, List<String>> dogFavoriteColors =
  new HashMap<String, List<String>>();

List<String> milouColors = dogFavoriteColors.get("Milou")
if(milouColors == null) {
  milouColors = new ArrayList<String>();
  dogFavoriteColors.put("Milou",milouColors);
}
milouColors.add("Vert");
```

< Multi Map >

Java 5

```
Map<String, List<String>> dogFavoriteColors =
  new HashMap<String, List<String>>();

List<String> milouColors = dogFavoriteColors.get("Milou")
if(milouColors == null) {
  milouColors = new ArrayList<String>();
  dogFavoriteColors.put("Milou",milouColors);
}
milouColors.add("Vert");
```

Guava

```
Multimap<String, String> dogFavoriteColors =
  HashMultimap.create();
dogFvoriteColors2.put("Milou", "Jaune");
dogFavoriteColors2.put("Milou", "Rouge");
```

< Bi Map >

Guava

```
BiMap<String, Dog> tatouages = HashBiMap.create();
tatouages.put("ABC123", new Dog("Milou") );
tatouages.put("MBD324", new Dog("Volt") );
tatouages.put("JFT672", new Dog("Lassie") );

println( tatouages );
```

{ABC123=Milou, MBD324=Volt, JFT672=Lassie}

```
println( tatouages.inverse() );
```

{Milou=ABC123, Volt=MBD324, Lassie=JFT672}

Une map bijective

Il est possible de changer la valeur associée à une clé mais pas d'avoir deux clés avec la même valeur (IllegalArgumentException).

< Multi Set >

Guava

```
Multiset<Integer> ages = HashMultiset.create();
ages.add(2);
ages.add(3);
ages.add(7);
ages.add(11);
ages.add(3);
ages.add(5);
println( ages );
```

[2, **3 x 2**, 5, 7, 11]

```
println( ages.count(3) )
```

2

Base

Préconditions, joiner, splitter, optional

- toString ()
- equals ()
- hashCode ()
- compareTo ()

< Preconditions >

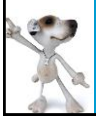
DOG

```
public class Dog {
  private Integer   id ;
  private String    name ;
  private String    fullName ;
  ...

  public Dog(String name, String fullName, ...) {
    if(name == null) {
      throw new NPE("Le nom bla bla");
    }
    if(fullName == null) {
      throw new NPE("bla bla");
    }
    ...
    this.name = name;
    this.fullName = fullName;
    ...
  }
```

< Preconditions >

DOG

```
import static com.google.common.base.Preconditions.*;
public class Dog {
  private Integer   id ;
  private String    name ;
  private String    fullName ;
  ...

  public Dog(String name, String fullName, ...) {
    this.name = checkNotNull(name, "bla bla");
    this.fullName = checkNotNull(fullName, "bla bla");
    ...
  }
```



checkNotNull() → NPE
checkArgument() → IAE
checkState() → ISE

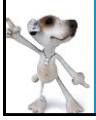
Joiner

```
String names = Joiner.on(", ")
  .join(newArrayList("Lassie", "Volt", "Milou"));
```

"Lassie, Volt, Milou"

Java classique

```
List<String> dogNames = newArrayList("Lassie", "Volt", "Milou");
StringBuilder sb = new StringBuilder();
boolean first = true;
for (String name : dogNames) {
  if(name != null || name.trim().isEmpty()) {
    continue;
  }
  if (!first) {
    sb.append(", ");
  }
  sb.append(name);
  first = false;
}
String names = sb.toString();
```



Joiner

```
String names = Joiner.on(", ")
  .join(newArrayList("Lassie", "Volt", "Milou"));
```
→ "Lassie, Volt, Milou"

```
String names = Joiner.on(", ")
  .join(newArrayList("Lassie", "Volt", null, "Milou"));
```
→ NPE ☹

Joiner

```
String names = Joiner.on(", ")
  .join(newArrayList("Lassie", "Volt", "Milou"));
```
→ "Lassie, Volt, Milou"

```
String names = Joiner.on(", ")
  .join(newArrayList("Lassie", "Volt", null, "Milou"));
```
→ NPE ☹

```
String names = Joiner.on(", ")
  .skipNulls()
  .join(newArrayList("Lassie", "Volt", null, "Milou"));
```
→ "Lassie, Volt, Milou"

Joiner

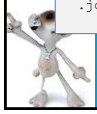
```
String names = Joiner.on(", ")
  .join(newArrayList("Lassie", "Volt", "Milou"));
```
→ "Lassie, Volt, Milou"

```
String names = Joiner.on(", ")
  .join(newArrayList("Lassie", "Volt", null, "Milou"));
```
→ NPE ☹

```
String names = Joiner.on(", ")
  .skipNulls()
  .join(newArrayList("Lassie", "Volt", null, "Milou"));
```
→ "Lassie, Volt, Milou"

```
String names = Joiner.on(", ")
  .useForNull("Anonymous"))
  .join(newArrayList("Lassie", "Volt", null, "Milou"));
```
→ "Lassie, Volt, Anonymous, Milou"

→ http://blog.developpez.com/guava/p11054/base/joiner-pour-assembler-des-items/



Splitter

```
Iterable<String> dogNames =
  Splitter.on(",")
          .split("Lassie, Volt, Milou");
```
→ ["Lassie", "Volt", "Milou"]

Splitter

```
Iterable<String> dogNames =
  Splitter.on(",")
          .split("Lassie, Volt, Milou");
```
→ ["Lassie", "Volt", "Milou"]

```
Iterable<String> dogNames =
  Splitter.on(",")
          .split("Lassie, Volt, ,Milou");
```
→ ["Lassie", "Volt", " ", "Milou"]

Splitter

```
Iterable<String> dogNames =
  Splitter.on(",")
          .split("Lassie, Volt, Milou");
```
→ ["Lassie", "Volt", "Milou"]

```
Iterable<String> dogNames =
  Splitter.on(",")
          .split("Lassie, Volt, ,Milou");
```
→ ["Lassie", "Volt", " ", "Milou"]

```
Iterable<String> dogNames =
  Splitter.on(",")
          .trimResults()
          .split("Lassie, Volt, ,Milou");
```
→ ["Lassie", "Volt", "", "Milou"]

Splitter

```
Iterable<String> dogNames =
  Splitter.on(",")
          .split("Lassie, Volt, Milou");
```
→ ["Lassie", "Volt", "Milou"]

```
Iterable<String> dogNames =
  Splitter.on(",")
          .split("Lassie, Volt, ,Milou");
```
→ ["Lassie", "Volt", " ", "Milou"]

```
Iterable<String> dogNames =
  Splitter.on(",")
          .trimResults()
          .split("Lassie, Volt, ,Milou");
```
→ ["Lassie", "Volt", "", "Milou"]

```
Iterable<String> dogNames =
  Splitter.on(",")
          .trimResults()
          .omitEmptyStrings()
          .split("Lassie, Volt, ,Milou");
```
→ ["Lassie", "Volt", "Milou"]

→ http://blog.developpez.com/guava/p11045/annotation/splitter-pour-separer-des-items/

Optional

Wrapper Optional <T>

```
Dog dog = new Dog("Milou", ...);
Optional<Dog> opt = Optional.of(dog);
assertTrue( opt.isPresent() );
assertEquals( "Milou", opt.get().getName() );
```

```
Optional<Dog> opt = Optional.absent();
assertFalse( opt.isPresent() );
opt.get(); → ISE
```

```
Dog dog = null;
Optional<Dog> opt = Optional.of(dog); → NPE
Optional<Dog> opt = Optional.fromNullable(dog);
```

```
Dog dog = null;
Optional<Dog> opt = Optional.fromNullable(dog);
Dog dog2 = opt.or( new Dog("noname", ...) );
assertEquals( "noname", dog2.getName() );
```

→ http://blog.developpez.com/guava/p11163/base/le-wrapper-optional-de-guava/

Functional Programming

todo

Super chien



Dog

Super Chien

```
public class SuperChien implements SuperHero {
  private String         surnom ;
  private double         poids ;
  private Set<String>    couleursCostume ;
  private Set<String>    pouvoirs ;
  ...
```

Les héros peuvent avoir plusieurs costumes donc je n'utilise pas un ImmutableSet. Idem pour les pouvoirs dont la liste augmente avec l'expérience.

< Transformation >

Transformation

```
List<SuperChien> superChiens =
  Lists.transform(dogs,
                  new Function<Dog, SuperChien>() {
    @Override
    public SuperChien apply(Dog dog) {
      SuperChien chien = new SuperChien();
      chien.setSurnom("Super " + dog.getName());
      chien.setPoids(dog.getWeight());
      chien.setCouleursCostume(newHashSet(dog.getColors()))
      chien.setPouvoirs(newHashSet("Code en Java", "Vole"))
      ...
      return chien;
    }
});
```

[Super Milou, Super Rintintin, Super Volt, Super Lassie, Super Pluto, Super Medor]

< Transformation >

Transformation

```
List<SuperChien> superChiens =
  Lists.transform(dogs, new Function<Dog, SuperChien>() {
    @Override
    public SuperChien apply(Dog dog) {
      SuperChien chien = new SuperChien();
      ...
      return chien;
```

- Vue (lazy)
- size / isEmpty dispo ☺
- Pas pour traitements répétés ☹ → FAQ 3

```
List<SuperChien> chiens = newArrayList(Lists.transform(...
```

```
ImmutableList<SuperChien> chiens =
                ImmutableList.copyOf(Lists.transform(...
```

< Filtre >

Filtre

```
Predicate<Dog> malePredicate =
  new Predicate<Dog>() {
    public boolean apply(Dog dog) {
      return dog.getSex() == MALE;
    }
  }
```

```
Iterable<Dog> maleDogs =
  Iterables.filter(dogs, malePredicate);
```

[Milou, Rintintin, Volt, Pluto, Medor]

< Filtre >

Filtre

```
Predicate<Dog> malePredicate =
  new Predicate<Dog>() {
    public boolean apply(Dog dog) {
      return dog.getSex() == MALE;
    }
  }
```

```
Iterable<Dog> maleDogs =
  Iterables.filter(dogs, malePredicate);
```

[Milou, Rintintin, Volt, Pluto, Medor]

```
Dog firstMaleDog =
  Iterables.find(dogs, malePredicate);
```

Milou

< Filtre >

Filtre

```
Predicate<Dog> malePredicate =
  new Predicate<Dog>() {
    public boolean apply(Dog dog) {
      return dog.getSex() == MALE;
    }
  }
```

```
Iterable<Dog> maleDogs =
  Iterables.filter(dogs, malePredicate);
```

[Milou, Rintintin, Volt, Pluto, Medor]

```
Dog firstMaleDog =
  Iterables.find(dogs, malePredicate);
```

Milou

```
Dog firstMaleDog =
  Iterables.find(femaleDogs, malePredicate, DEFAULT_DOG );
```

Default dog

Seconde liste

```
List<Dog> dogs2 = newArrayList(
  new Dog("Rintintin", 45.0, MALE, ...),
  new Dog("Pluto", 22.0, MALE, ...),
  new Dog("Lulu", 35.6, MALE, ...));
```

[Rintintin, Pluto, Lulu]

Liste 2 : dogs2

[Milou, **Rintintin,** Volt, Lassie, **Pluto,** Medor]

Liste 1 : dogs

< Fluent or not fluent ? >

Pas fluent

```
Predicate<Dog> malePredicate =
  new Predicate<Dog>() {
    public boolean apply(Dog dog) {
      return dog.getSex() == MALE;
    }};
```

```
Function<FullDog, String> nameFunction =
  new Function<FullDog, String>() {
    public String apply(FullDog dog) {
      return dog.getName();
    }};
```

```
Iterable<FullDog> maleDogs =
  Iterables.filter(dogs, malePredicate);
Iterable<String> maleNames =
  Iterables.transform(maleDogs, nameFunction);
```

[Milou, Rintintin, Volt, Pluto, Medor]

< Fluent or not fluent ? >

Pas fluent

```
Iterable<FullDog> maleDogs =
  Iterables.filter(dogs, malePredicate);
Iterable<String> maleNames =
  Iterables.transform(maleDogs, nameFunction);
```

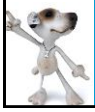
Fluent

```
List<String> maleNames2 =
  FluentIterable.from(dogs)
    .filter(malePredicate)
    .transform(nameFunction)
    .toImmutableList();
```

Guava 12.0

[Milou, Rintintin, Volt, Pluto, Medor]

→ http://blog.developpez.com/guava/p11092/annotation/fluentiterable-sur-mon-chien-guava/



Cache

todo

< Memoization >

Web Service

```
public class DogService {
  @Inject
  private PetShopWebService service;
  public Integer getNumberDogsSoldYesterday() {
    return service.checkSales( "dog" );
  }
```

< Memoization >

Web Service

```
public class DogService {
  @Inject
  private PetShopWebService service;
  public Integer getNumberDogsSoldYesterday() {
    return service.checkSales( "dog" );
  }
```

Cache manuel

```
  private Integer nbOfDogsSold;
  public Integer getNumberDogsSoldYesterday() {
    if (nbOfDogsSold == null) {
      nbOfDogsSold = service.checkSales( "dog" );
    }
    return nbOfDogsSold;
  }
```

Double check null...

< Memoization >

Guava

```
public class DogService {
  @Inject
  private PetShopWebService service;
  private Supplier<Integer> nbOfDogsSoldSupplier =
    Suppliers.memoize(
      new Supplier<Integer>() {
        public Integer get() {
          return service.checkSales( "dog" );
        }
      });
  public Integer getNumberDogsSoldYesterday() {
    return nbOfDogsSoldSupplier.get();
  }
```

< Memoization >

Guava

```
public class DogService {
  @Inject
  private PetShopWebService service;
  private Supplier<Integer> nbOfDogsSoldSupplier =
    Suppliers.memoizeWithExpiration(
      new Supplier<Integer>() {
        public Integer get() {
          return service.checkSales( "dog" );
        }
      }, 1, TimeUnit.DAYS );
  public Integer getNumberDogsSoldYesterday() {
    return nbOfDogsSoldSupplier.get();
  }
```

< Cache >

Web Service

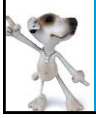
```
public class DogService {
  @Inject
  private PetShopWebService service;
  private Map<String, Dog> dogMap = Maps.newHashMap();
  public Integer getDog(String name) {
    Dog dog = dogMap.get(name);
    if(dog == null) {
      dog = service.getAnimal( "dog", name ); // type-name
      dogMap.put( name, dog );
    }
    return dog;
  }
```

Quid du timeout ? Max ?

< Cache >

Guava

```
public class DogService {
  @Inject
  private PetShopWebService service;
  private LoadingCache<String, Dog> dogCache =
    CacheBuilder.newBuilder()
                .maximumSize(2000)
                .expireAfterWrite(30, TimeUnit.MINUTES)
                .build(new CacheLoader<String, Dog>() {
      public Dog load(String key) {
        return service.getAnimal( "dog", key );
      }
    });
  public Integer getDog(String name) {
    return dogCache.get( name ); // + try-catch
  }
```



Hash

Fantôme

Hash

```
HashFunction hf = Hashing.md5();
HashCode hc = hf.newHasher()
                .putInt(123)
                .putString("Milou")
                .hash();
byte[] bytes = hc.asBytes();
```

- md5
- Murmur3 128 bits
- Murmur3 32 bits
- Sha1
- Sha256
- Sha512
- *goodFastHash*

Préparation

< Is in list ? >

```
int NB_OF_DOGS = 100000;

List<Dog> dogs = newArrayList();
Random rand = new Random();
for (int i = 0; i < NB_OF_DOGS; i++) {
  Dog dog = new Dog();
  dog.setName("abc" + rand.nextInt(999));
  ...
  dogs.add(dog);
}
```

contains

```
final Dog milou = new Dog();
milou.setName("Milou");
...
boolean isInList = dogs.contains(milou);
```

false (14 ms)

```
dogs.add(milou);
boolean isInList = dogs.contains(milou);
```

true (14 ms)

Bloom filter

< Is in list ? >

```
Funnel<Dog> dogFunnel = new Funnel<Dog>() {
  public void funnel(Dogdog, PrimitiveSink sink) {
    sink.putString(dog.getName())
        .putString(dog.getFullName())
        .putString(dog.getRace());
  }};
BloomFilter<Dog> bloom =
  BloomFilter.create(dogFunnel, NB_OF_DOGS, 0.01);
for (int i = 0; i < NB_OF_DOGS; i++) {
  ...
  bloom.put(dog);
```

```
boolean isInList = bloom.mightContain(milou);
```

false (0 ms)

```
bloom.put(milou);
boolean isInList = bloom.mightContain(milou);
```

true (0 ms)

Guava 13

→ http://blog.developpez.com/guava/p11149/collection/bloom-filter-de-guava-13/

Guava or not Guava ?

Avantages et inconvénients

Pro / Cons

- Ne pas en abuser…
- Utile
- Bonnes pratiques

LIENS

@thierryleriche

Guava
→ http://code.google.com/p/guava-libraries

Lombok
→ http://projectlombok.org

Lombok-pg
→ https://github.com/peichhorn/lombok-pg

ICAUDA
→ http://icauda.com
→ http://icauda.com/articles.html *(articles)*
→ http://icauda.com/cours.html *(slides)*

Blog Guava
→ http://blog.developpez.com/guava

« Simplifier le code de vos beans Java à l'aide de Commons Lang, Guava et Lombok »
→ http://thierry-leriche-dessirier.developpez.com/tutoriels/java/simplifier-code-guava-lombok *(article)*

MERCI

FAQ / Bonus

1. Guava Vs Commons
2. Régime façon Apache
3. Fonctionnal prog Vs boucle for
4. Créer des Maps
5. Charsets
6. Converter Spring
7. Orderings
8. Licences
9. Extension avec valeurs
10. Chrono
11. CharMatcher
12. Impl
13. And, or, in…
14. Partition, limit

Guava Vs Commons ?

http://tinyurl.com/guava-vs-apache

1

Régime : la méthode Commons ?

toString

```
public String toString() {
   return new ToStringBuilder(this)
                .append("id", id)
                .append("name", name)
                .append("fullName", fullName)
                ...
                .append("colors", colors)
                .toString();
}
```

2

Régime : la méthode Commons ?

equals

```
public boolean equals(Object obj) {
  if (this == obj) return true;
  if (obj == null) return false;
  if (!(obj instanceof Dog)) return false;

  Dog other = (Dog) obj;

  return new EqualsBuilder()
              .append(birthday, other.birthday)
              .append(fullname, other.fullname)
              .append(name, other.name)
              .append(race, other.race)
              .append(sex, other.sex)
              .isEquals();
}
```

Régime : la méthode Commons ?

hashCode

```
public int hashCode() {
  return new HashCodeBuilder(17, 37)
              .append(birthday)
              .append(fullname)
              .append(name)
              .append(race)
              .append(sex)
              .toHashCode();
}
```

Régime : la méthode Commons ?

compareTo

```
public int compareTo(Dog other) {
  return new CompareToBuilder()
              .append(name, other.name)
              .append(fullname, other.fullname)
              .append(birthday, other.birthday)
              ...
              .append(sex, other.sex)
              .toComparison();
}
```

Fonctionnal prog Vs boucle for : quand ?

Transformation

```
List<SuperChien> superChiens = newArrayList(
  Lists.transform(dogs, new Function<Dog, SuperChien>() {
    @Override
    public SuperChien apply(Dog dog) {
      SuperChien chien = new SuperChien();
      ...
      return chien;
    }
}));
```

Vs

```
List<SuperChien> superChiens = newArrayList();
for(Dog dog : dogs) {
  SuperChien chien = new SuperChien();
  ...
  superChiens.add(chien);
}
```

http://code.google.com/p/guava-libraries/wiki/FunctionalExplained

Créer des Maps

of

```
public static final ImmutableMap<String, Integer>
  AGES = ImmutableMap.of("Milou", 32,
                         "Volt", 7,
                         "Pluto", 37,
                         "Lassie", 17);
```

Builder

```
public static final ImmutableMap<String, Integer>
  AGES = new ImmutableMap.Builder<String, Integer>()
           .put("Milou", 32)
           .put("Volt", 7)
           .put("Pluto", 37)
           .put("Lassie", 17)
           .put("Medor", 5)
           .put("Croquette", 8)
           .put("Loulou", 2)
           ...
           .build();
```

Charsets

Java 5

```
String name = "Milou";
try {
  byte[] bytes = name.getBytes("UTF-8");
} catch (UnsupportedEncodingException e) {
  throw new AssertionError(e);
}
```

Ca n'arrive jamais cette exception ☹

Guava

```
String name = "Milou";
byte[] bytes = name.getBytes(Charsets.UTF_8);
```

Converter Spring

```
import org.springframework...Converter;

@Component("dogToChienConverter")
public class DogToChienConverter
                implements Converter<List<Dog>,
                                     List<SuperChien>> {

  public List<SuperChien> convert(List<Dog> dogs) {
    List<SuperChien> chiens = newArrayList(transform(dogs,
      new Function<Dog, SuperChien>() {
        public SuperChien apply(Dog dog) {
          ...
          return chien;
        }
    }));
    return chiens;
  }
```

Lazy or not lazy ?

Orderings

Licences

- Lombok : MIT License
- Lombok-pg : MIT License
- Guava : Apache License 2.0
- Commons : Apache License 2.0

Extension avec valeurs

```
@ExtensionMethod({ Object.class, MyOtherClass.class })
public class DogWithExtension {

  public void foo() {
    String s1 = "toto";
    s1.print();              // → toto

    String s2 = null;
    s2.print();              // → null
  }
}

class MyOtherClass {
  public static <T> void print(final T value) {
    System.out.println(value);
  }
```

On peut mettre autre chose qu'un "Object", par exemple un "Arrays" , un "Dog" , etc.

Extension avec valeurs

```
@ExtensionMethod({ Object.class, MyOtherClass.class })
public class DogWithExtension {

  public void foo() {
    String s1 = "toto";
    s1.print();              // → toto

    String s2 = null;
    s2.print();              // → null
    s2.print("vide");       // → vide
  }
}

class MyOtherClass {
  public static <T> void print(final T value) {
    System.out.println(value);
  }

  public static void print(String value, String other) {
    if (value == null || value.isEmpty()) {
      System.out.println(other);
    } else {
      System.out.println(value);
    }
```

Guava Java 5

< Chrono >

```
long start = new Date().getTime();
foo(); // traitement long (ou pas)
long end = new Date().getTime();
long duration = end - start; // 11 ms
```

```
Stopwatch sw = new Stopwatch();
sw.start();
foo();
sw.stop();
long duration = sw.elapsedMillis(); // 11 ms
```

```
long nano = sw.elapsedTime(NANOSECONDS); // 11179739 ns
long micro = sw.elapsedTime(MICROSECONDS); // 11179 us
long millis = sw.elapsedTime(MILLISECONDS); // 11 ms
```

→ http://blog.developpez.com/guava/p11160/base/le-stop-watch-de-guava/

CharMatcher

11

Impl

```
public class Dog implements Comparable<Dog> {
  private Integer id ;
  private String  name ;
  private String  fullName ;
  ...
  @Override
  public int compareTo(Dog dog) {
    return ...;
  }
```

5

12

implements List ? aie aie aie

Impl

```
@AutoGenMethodStub
public class Dog implements Comparable<Dog> {
  private Integer id ;
  private String  name ;
  private String  fullName ;
  ...
```

L

12

And, or, in…

And, or, in…

```
import static com.google.common.base.Predicates.and;
import static com.google.common.base.Predicates.in;
boolean isRintintinInBoth =
  and( in(dogs), in(dogs2) )
    .apply(new Dog("Rintintin"));
```

G

true

```
import static com.google.common.base.Predicates.or;
boolean isTintinInOne =
  or( in(dogs), in(dogs2) )
    .apply(new Dog("Tintin");
```

G

false

13

Liste 1 : dogs

[Milou, **Rintintin**, Volt, Lassie, **Pluto**, Medor]

Liste 2 : dogs2

[**Rintintin**, **Pluto**, Lulu]

Partition, limit

Partition

```
List<List<FullDog>> partition =
  Lists.partition(dogs, 4);
```

G

[Milou, Rintintin, Volt, Lassie]

[Pluto, Medor]

Limit

```
List<FullDog> first4Dogs =
  newArrayList(Iterables.limit(dogs, 4));
```

G

[Milou, Rintintin, Volt, Lassie]

14

Slides en préparation

I/O

todo

< I/O >

Ressources

```
public void lireJava6(String from) {
  InputStream in = null;
  try {
    in = new FileInputStream(from);
    byte[] b = new byte[10000];
    while (true) {
      int r = in.read(b);
      if (r == -1)
        break;
      ...
    }
  } catch (Exception e) {
    ...
  } finally {
    if (in != null) {
      try {
        in.close();
      } catch (Exception e) {
        ...
      }
    }
  ...
```

< I/O >

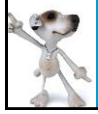
Java 7

```
public void lireJava7(String from) {
  try(InputStream in = new FileInputStream(from)) {
    ...
    while (true) {
      ...
    }
  }
}
```


< I/O >

Guava

Todo



Concurrent / future

todo

Todo