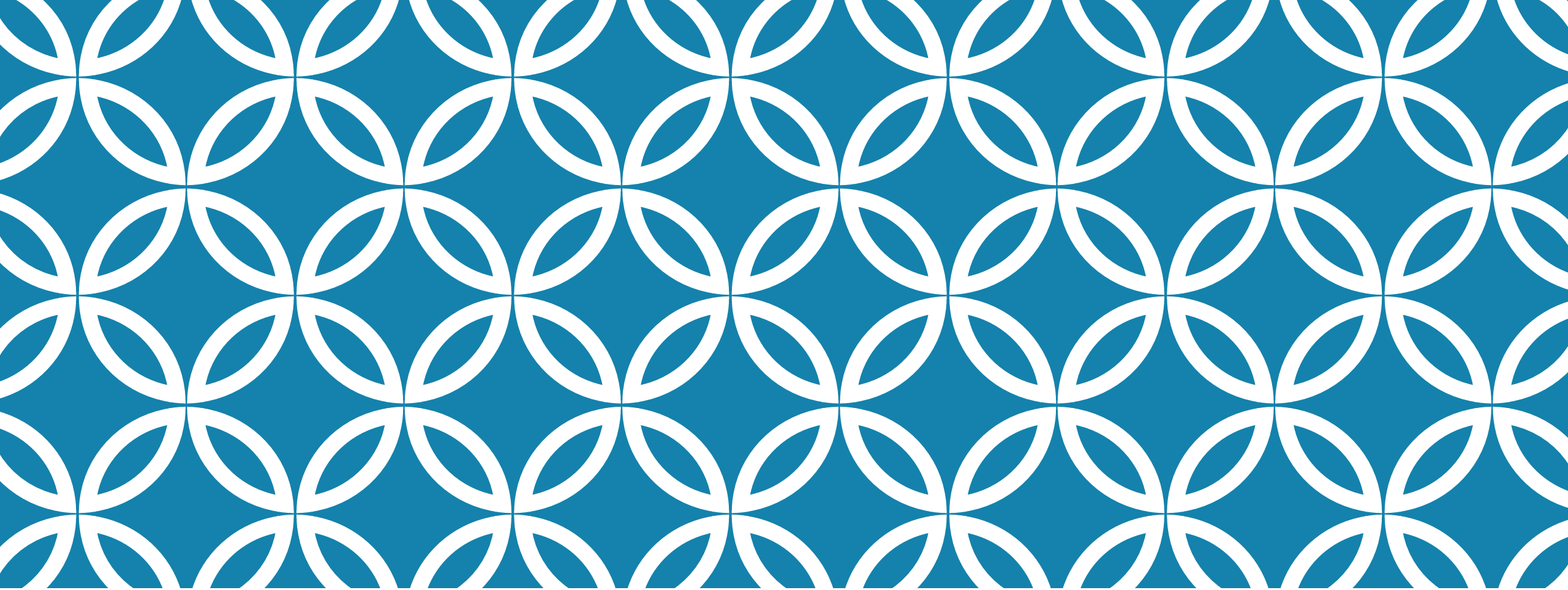
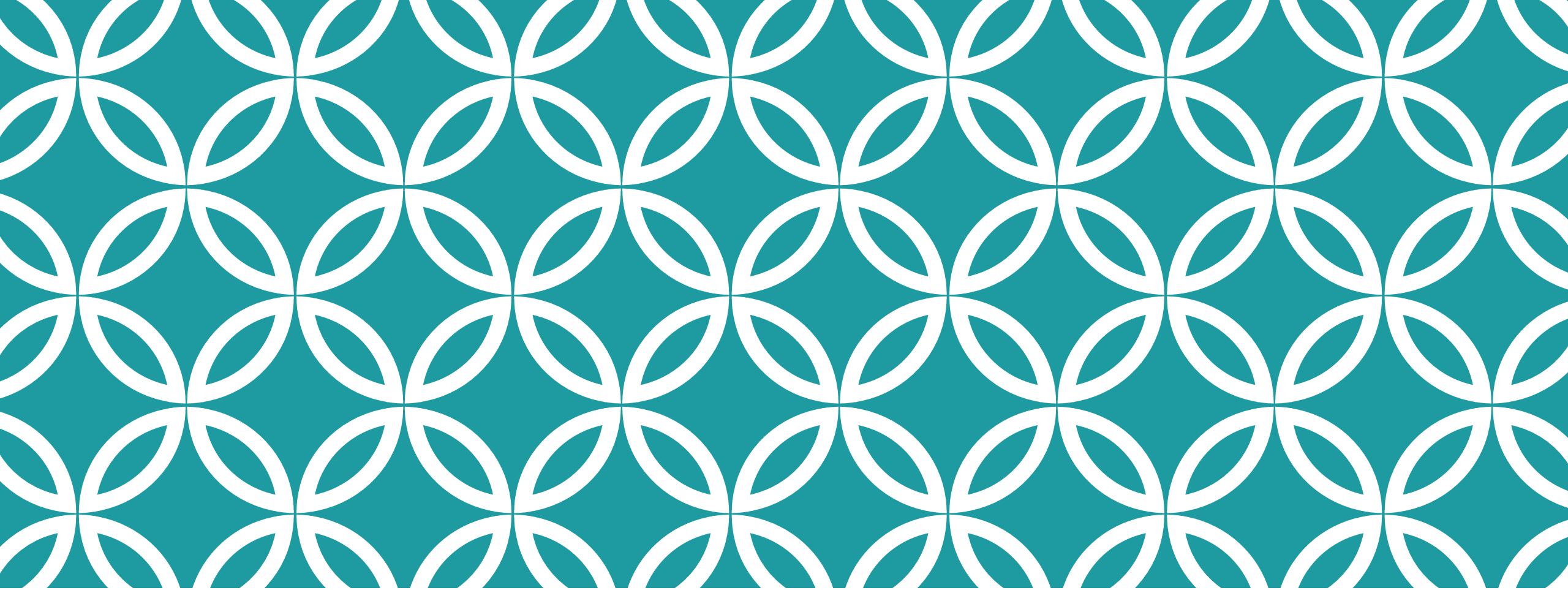


PROGRAMMATION ORIENTÉE OBJET

Jérémy PERROUAULT



PRÉSENTATION OBJET

Introduction POO

L'OBJET

POO	Réalité
Classe	Plans pour fabriquer un smartphone
Objet (instance de classe)	Le smartphone
Propriétés	Attributs du smartphone (couleur, poids, dimensions, …)
Méthodes	Fonctionnalités du smartphone (téléphoner, SMS)

L'OBJET (EXERCICE)

POO	Réalité
?	Jérémy
?	Marcher, courir, pleurer, crier, gagnerExperience
?	ADN Humain
?	Couleur peau, couleur des yeux, taille

L'OBJET

Objet	Classe
Objet	Instance de classe
Type d'objet	Le nom de la classe

L'OBJET

Le smartphone est composé d'autres objets

- Ecran tactile
- Antenne
  - Antenne wifi
  - Antenne télécom
- Connecteur Jack
- …

L'OBJET (EXERCICE)

Modéliser un chat (simplement)

- Définir ses propriétés
- Définir ses méthodes

L'OBJET

Un objet, c'est donc la modélisation

- D'une chose tangible (Smartphone, Stylo, Voiture, Personne, …)
- D'une chose conceptuelle (Réunion, Service, Idée, …)

Une classe, c'est donc le modèle de l'objet

- Un objet, c'est l'instance d'une classe

LA PORTÉE

Les propriétés et les méthodes peuvent être

- Publiques — Accessibles par tous les autres objets
- Privées — Accessibles uniquement par l'objet lui-même (fermées aux autres)
- Protégées — Accessibles par l'objet et ses classes « filles » (fermées aux autres)
- Package — Accessibles par les objets du même package (par défaut en JAVA)

L'ENCAPSULATION

Principe de ne pas avoir besoin de savoir comment ça fonctionne

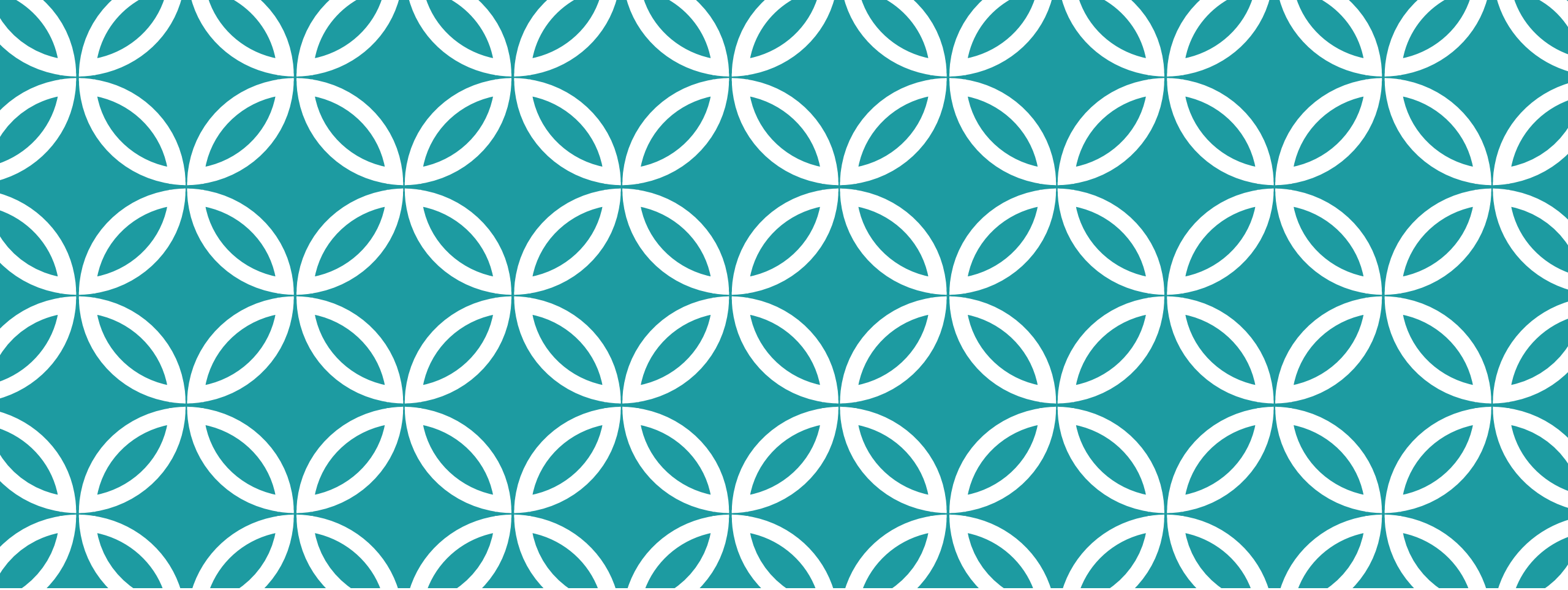
- Une interface masque la complexité

Un Smartphone ne présente qu'une interface Homme/Machine

- Nous manipulons le Smartphone, et lui interagit avec les autres objets et éléments qui le composent

Permet également de protéger l'information contenue dans l'objet

- Ne présente que les actions possibles, ou les propriétés



PROGRAMMATION ORIENTÉE OBJET

La POO

APPLICATION

Programmation procédurale

- Définition de fonctions
- Dirigée par les traitements
- Processus
  - Point d'entrée unique (traitement)
  - En fonction des cas : appelle un traitement ou un autre

Programmation Orientée Objet

- Définition d'objets qui interagissent entre eux
- Dirigée par les données
- Processus
  - Point d'entrée unique (objet)
  - Instancie d'autres objets et les utilise

Meilleure séparation des données et des fonctions qui les manipulent

APPLICATION

Réutilisabilité

Maintenance facilitée

Evolutivité accrue

APPLICATION

Chaque instance a une vie dans l'application

- Des modifications sur l'un n'aura pas d'effet sur les autres objets

Distinction par son adresse en mémoire

- Deux objets « Smartphone » qui auraient les mêmes caractéristiques
- Ils sont identiques sur le papier, mais sont réellement deux objets bien différents
  - Ils ont chacun une adresse mémoire qui leur est propre
- C'est un peu l'équivalent d'une « clé primaire » dans une base de données

APPLICATION

Un objet est une classe qui peut être créée et détruite

- Constructeur
  - Méthode appelée à l'instanciation de l'objet en mémoire
- Destructeur
  - Méthode appelée avant sa destruction en mémoire

Un objet n'est détruit que d'une seule façon

Un objet peut être construit de différentes façons

- Plusieurs constructeurs
- Plusieurs paramètres

APPLICATION

Le constructeur est un méthode implicite

- Pas obligatoire de le décrire dans la classe
- ATTENTION : Si un constructeur avec paramètres est défini, le constructeur par défaut n'existe plus de façon implicite : il faut le décrire de nouveau

DÉFINITION EN CODE

JAVA

```java
public class Chat {
  public String attributPublic;
  private String attributPrive;

  public Chat() {
    //...
  }


  public void manger() {
    //...
  }
}
```

UTILISATION EN CODE

Pour utiliser une classe, il faut l'instancier

Nom de la variable

- `Chat monChat = new Chat()`

Nom de la classe

Instanciation de la classe Chat

L'objet de type Chat sera sauvegardé dans la variable monChat

Appeler (exécuter) la méthode "manger"

- `monChat.manger()`

Nom de la variable

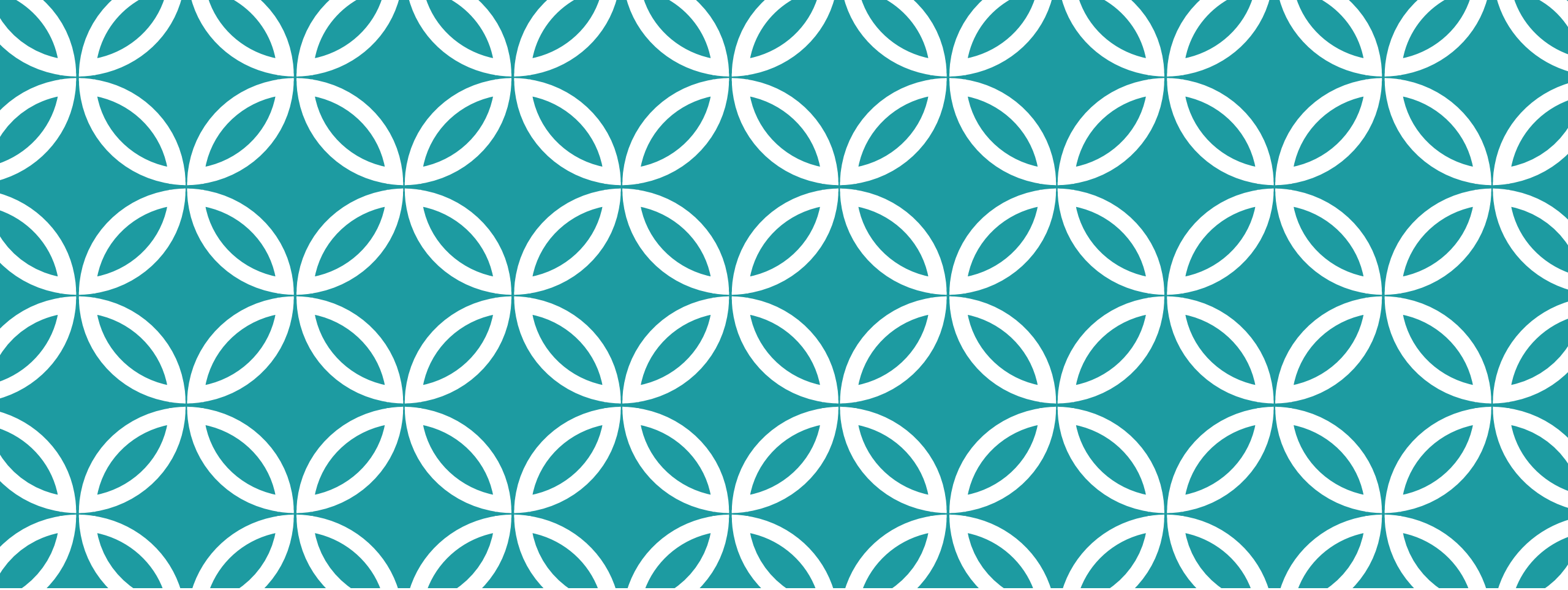
La méthode "manger" de monChat sera appelée, pas la méthode d'un autre chat

EXERCICE

Créer une classe « Chat »

- Avec un constructeur
  - Qui affiche dans la console « Création d'un chat ! »
  - Qui n'attend pas de paramètre
  - Puis qui attend le nom du chat en paramètre et qui affiche « Création du chat 'nom du chat' ! »

Tester dans le programme principal



CONVENTIONS

Les règles communes

NOM DES CLASSES

De manière générale, on va nommer les classes

- Première lettre en majuscule
- Si composée de plusieurs mots : première lettre de chaque mot en majuscule

Chat

ChatSansGriffes

PROPRIÉTÉS ET MÉTHODES

De manière générale, en JAVA

- Première lettre en minuscule
- Si composée de plusieurs mots : première lettre de chaque mot en majuscule

- Convention de nommage appelée « lowerCamelCase »

chat

monChatQuiDort

NOM DES INTERFACES

De manière générale, on va nommer les interfaces

- Première lettre « I » (comme Interface) en majuscule
- Lettre suivante en majuscule
- Si composée de plusieurs mots : première lettre de chaque mot en majuscule

IChasseur

IChasseurPuissant

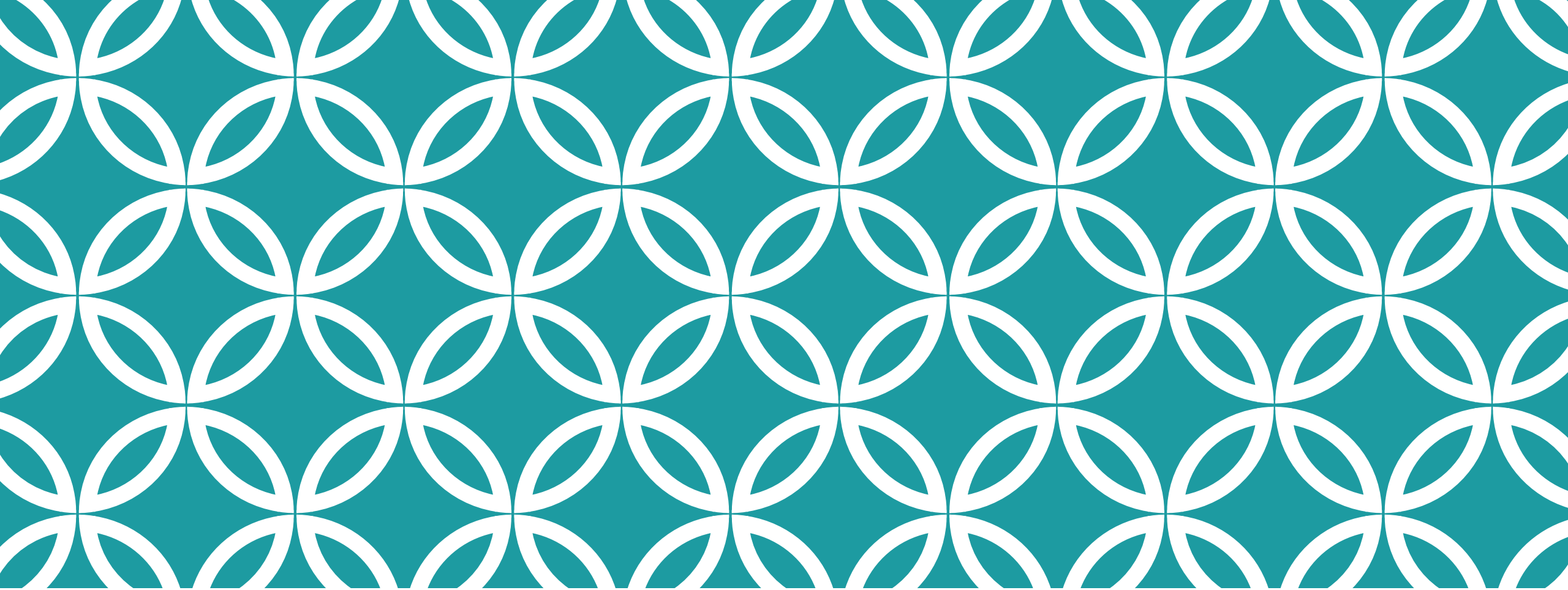
PROPRIÉTÉS

Une propriété doit toujours être privée ou protégée, jamais publique

Il faut des méthodes d'accès en lecture et en écriture publiques

- Accesseurs (getters, setters)

- maPropriete
- getMaPropriete()
- setMaPropriete(valeur)



RELATIONS ENTRE CLASSES

Les relations
L'héritage

HÉRITAGE

Principe visant à favoriser la réutilisabilité et l'adaptabilité des objets

Très proche de la phylogénétique

- Liens de parenté
  - Un chat est un animal, mais un animal n'est pas un chat

Classe « fille »

- Classe qui hérite d'une autre classe (on dit aussi qu'elle étend les fonctionnalités, ou spécifie)

Classe « mère »

- Classe directement parente de la classe « fille »

HÉRITAGE

En JAVA, on manifeste la notion d'héritage via le mot-clé

- extends

Dans un langage orienté objet, tout est Object, tout hérite naturellement de Object

- Object est la plus « haute » classe en POO

HÉRITAGE

« fille » possède toutes les propriétés et méthodes de sa « mère »

- Publiques et protégées, elle n'aura pas accès aux éléments privés

« fille » spécialise les propriétés et les méthodes de sa « mère »

- Un chat, c'est un mammifère plus spécifié, qui lui-même est moins abstrait qu'un animal
  - Spécialisation de la méthode "manger"
  - Un chat ne mange pas de la même façon qu'un chimpanzé
  - C'est donc la méthode "manger" du chat qui sera utilisée, et non la méthode de l'animal
    - Polymorphisme d'héritage
- Un chat peut miauler, tandis qu'un chien peut aboyer

POLYMORPHISME

Il existe plusieurs types de polymorphismes

- Héritage (overriding)
  - C'est la réécriture – la spécialisation – d'une méthode
  - Un chat a sa propre façon de manger, mais tous les mammifères mangent

- Ad hoc (overloading)
  - C'est la redéfinition d'une méthode dans la même classe, avec d'autres paramètres
  - Un chat court différemment selon le type de terrain (terre ou boue)

HÉRITAGE

Le polymorphisme d’héritage, c’est la réécriture – la spécialisation – d’une méthode

EXERCICE

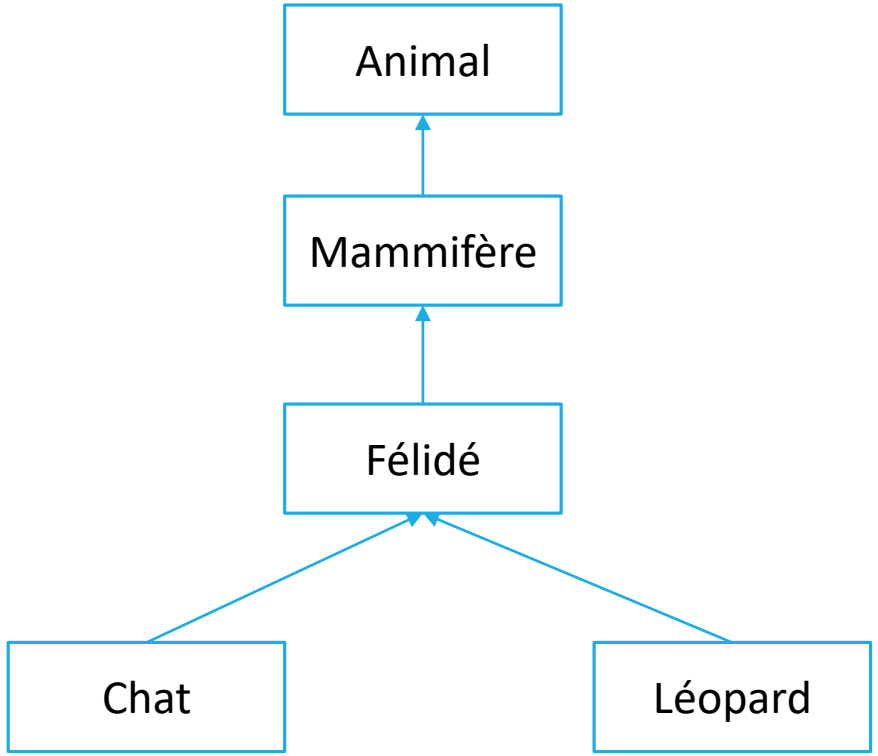
Créer une classe « Felide »

- La classe Chat doit en hériter

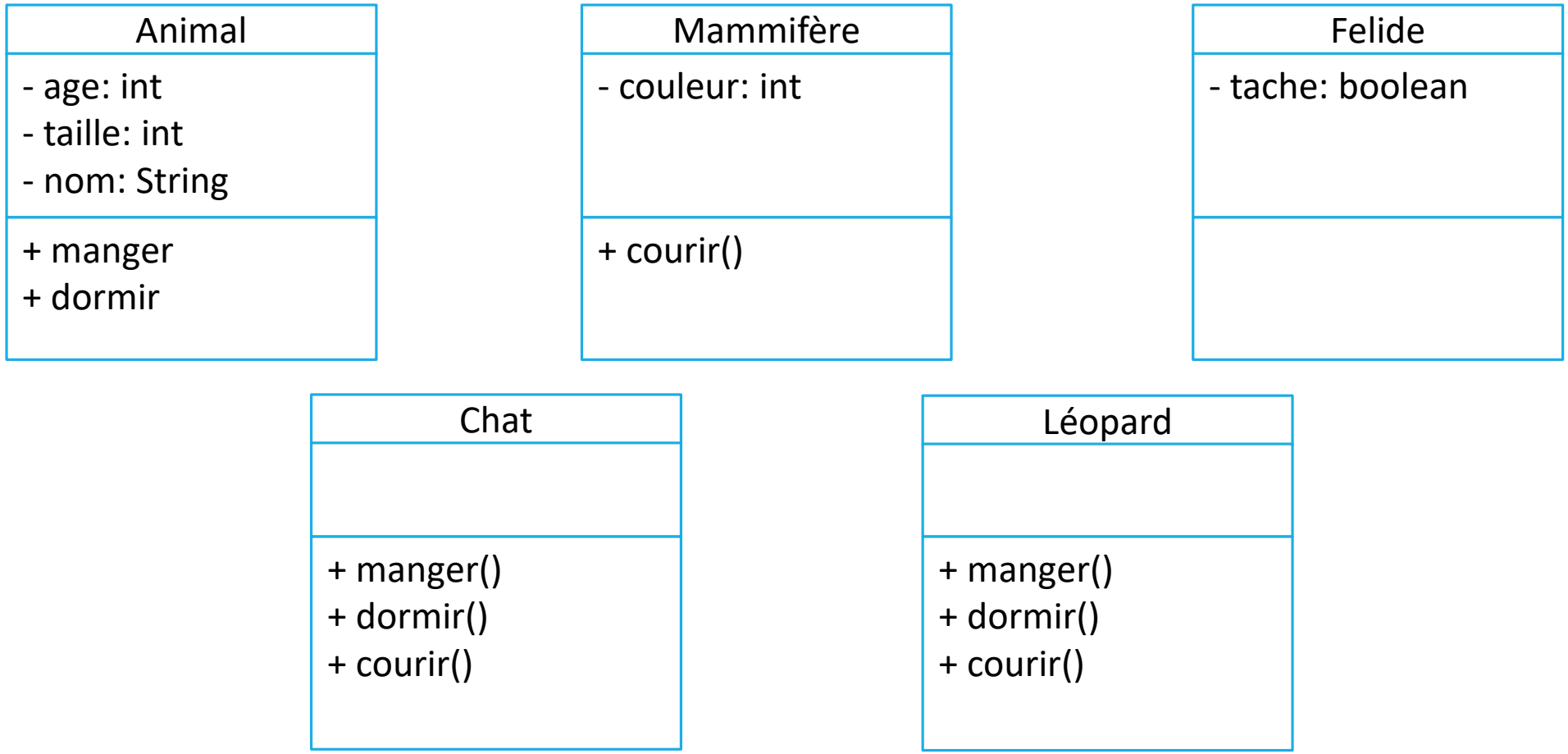
- Ajouter un constructeur à Felide
  - Afficher « Création d'un Félidé ! »
  - Que se passe-t-il ?

EXERCICE

Modéliser sous forme de classes le schéma suivant



EXERCICE


Animal
- age: int
- taille: int
- nom: String
+ manger
+ dormir
Mammifère
- couleur: int
+ courir()
Felide
- tache: boolean
Chat
+ manger()
+ dormir()
+ courir()
Léopard
+ manger()
+ dormir()
+ courir()