XDNATIS

OOP

Comprendre la Programmation Orientée Objet

XWIKI SAS

XDNATIS

Compétence demandée :

Comprendre ce qu'est une classe et les 4 principes de la POO

XDNATIS

1. Les classes
2. L’intanciation
3. L’abstraction
4. L’héritage
5. Le polymorphisme
6. L’encapsulation

XEBIA NATIS

1. Les classes
2. L’intanciation
3. L’abstraction
4. L’héritage
5. Le polymorphisme
6. L’encapsulation

XWIKI NATIS

LES CLASSES

XONATIS

XWIKI NATIS

La définition d'un objet
Versus
L'objet concret

XWIKI NATIS

La classe d'un objet
Versus
L'objet

XEBIA NATIS

Une classe regroupe des membres,
communs à un ensemble d'objets.

Ces membres peuvent être des
méthodes ou des propriétés

XWIKI NATIS

Les propriétés définissent les caractérisques d'un ensemble d'objets

Les méthodes définissent les comportements d'un ensemble d'objets

XONATIS

XWIKI SAS

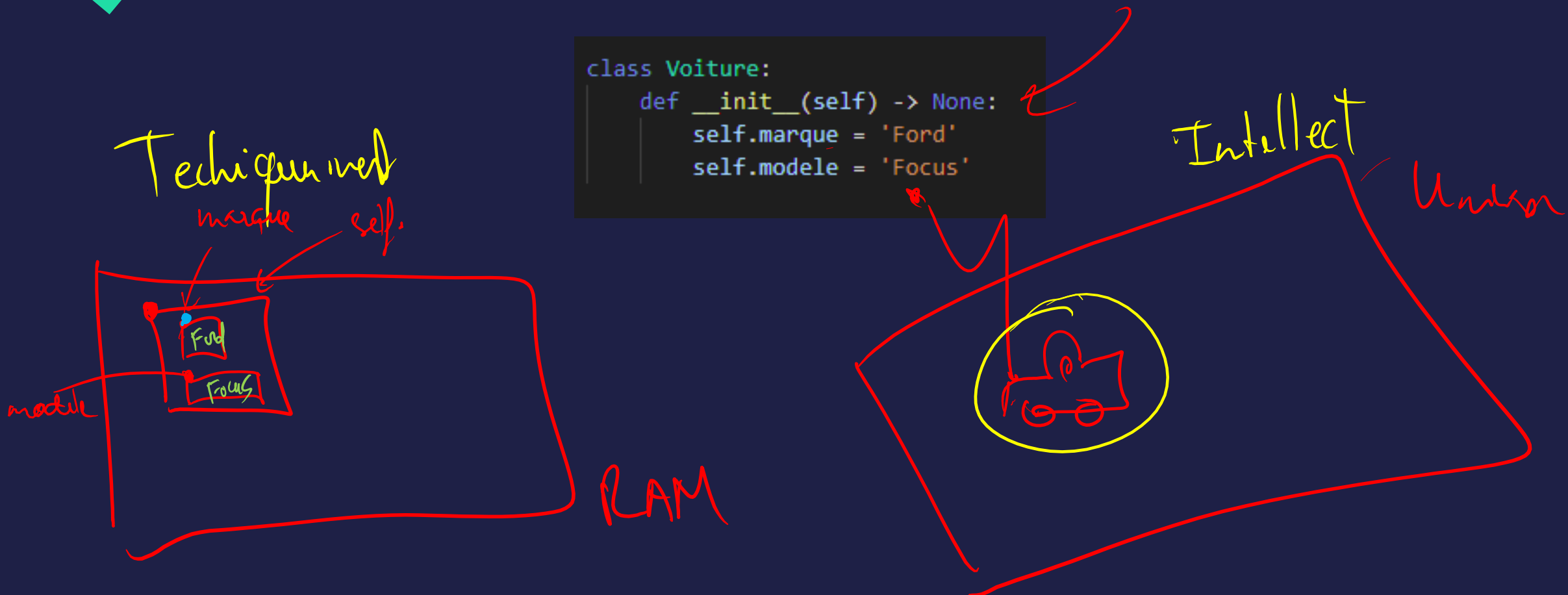
Exemples **en programmation**

XEBIANATIS

self.marque

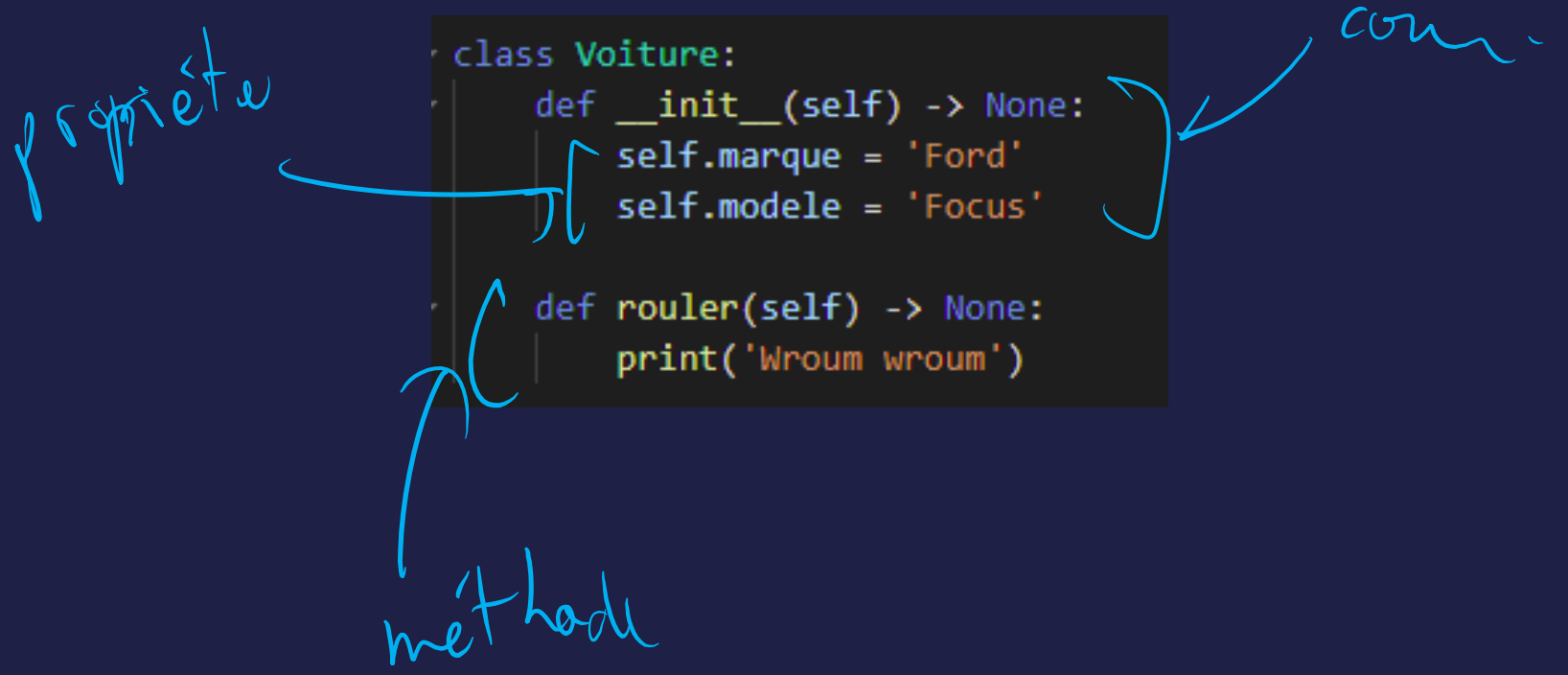
Les classes peuvent aussi définir des propriétés

```python
class Voiture:
    def __init__(self) -> None:
        self.marque = 'Ford'
        self.modele = 'Focus'
```



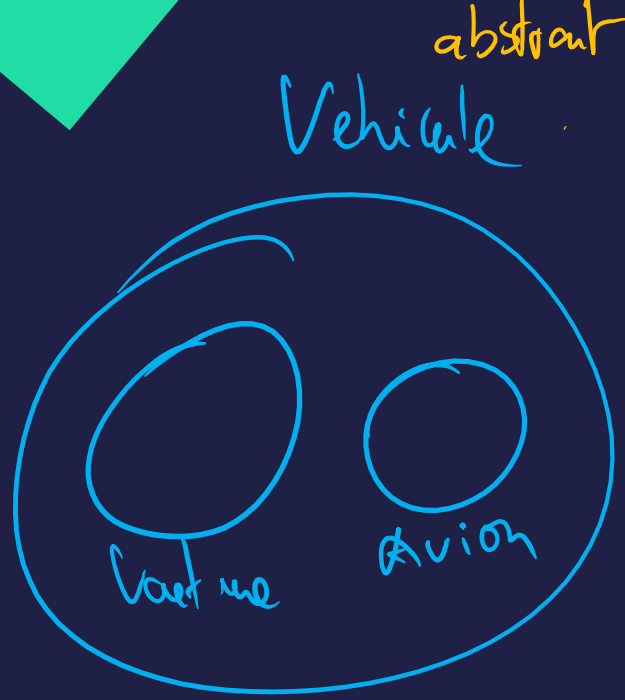
XNATIS

Les classes peuvent aussi définir des méthodes



XDNATIS

Les classes peuvent être abstraites



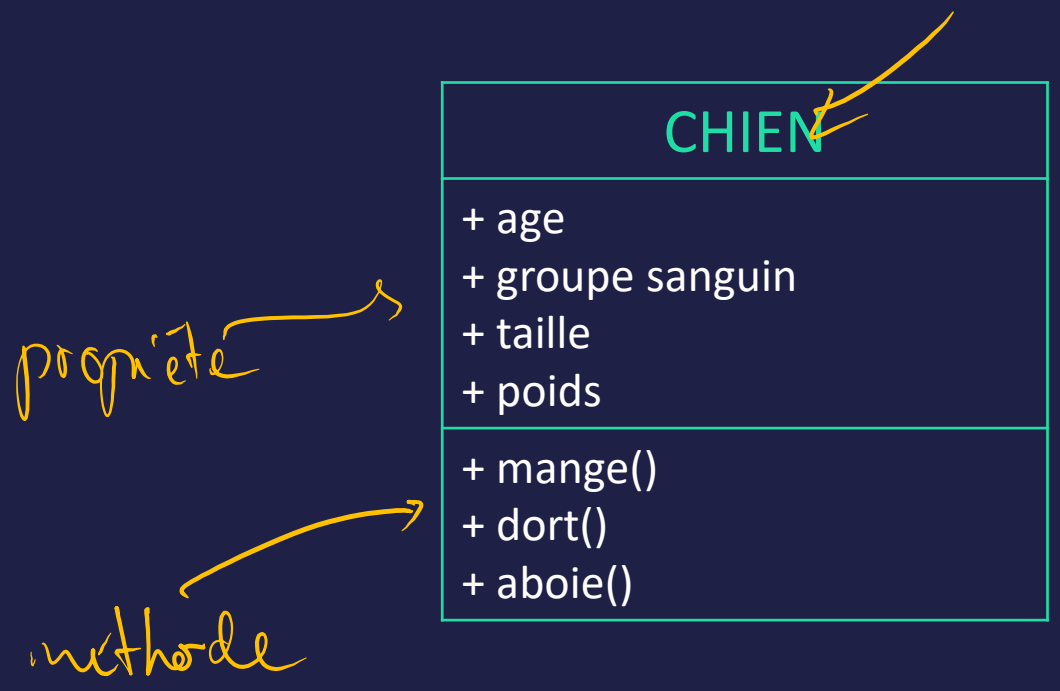

```python
from abc import ABC, abstractmethod

class Vehicule(ABC):
    @abstractmethod
    def transporter(self) -> None:
        pass

class Voiture(Vehicule):
    def transporter(self) -> None:
        print('Je roule')

class Avion(Vehicule):
    def transporter(self) -> None:
        print('Je vole')
```

XWIKI SAS

XDNATIS

Exemples **en UML**

XNATIS


CHIEN
+ age
+ groupe sanguin
+ taille
+ poids
+ mange()
+ dort()
+ aboie()
propriété
méthode

XONATIS

XEBIA NATIS

1. Les classes
2. L’intanciation
3. L’abstraction
4. L’héritage
5. Le polymorphisme
6. L’encapsulation

XWIKI NATIS

L'INSTANCIATION

XWIKI NATIS

Et dans un ordinateur ?

Que se passe-t-il dans la RAM ?

XWIKI SAS

XWIKI SAS

XEBIA NATIS

1. Les classes
2. L’intanciation
3. L’abstraction
4. L’héritage
5. Le polymorphisme
6. L’encapsulation

XWIKI NATIS

ABSTRACTION

XWIKI NATIS

Sinon … on s’est fiche ..

XNATIS

Touche de clavier d'ordinateur

XNATIS

Une batterie d’ordinateur

XWIKI NATIS

Une souris d'ordinateur

XNATIS

Un vidéo-projecteur

XDNATIS

Des hauts-parleurs

XWIKI NATIS

Un microphone

XWIKI NATIS

Une pédale de frein

XWIKI SAS

XWIKI NATIS

Abstraction

Utiliser des briques sans connaître leurs détails techniques

XWIKI NATIS

Abstraction

Utiliser des briques sans connaître leurs détails techniques

XWIKI SAS

XEBIA NATIS

1. Les classes
2. L’intanciation
3. L’abstraction
4. L’héritage
5. Le polymorphisme
6. L’encapsulation

XWIKI NATIS

L'HERITAGE

XWNATIS

CHIEN
+ age + groupe sanguin + taille + poids
+ mange() + dort() + aboie()

CHAT
- Coussinets + age + groupe sanguin + taille + poids
+ mange() + dort() + miauler()

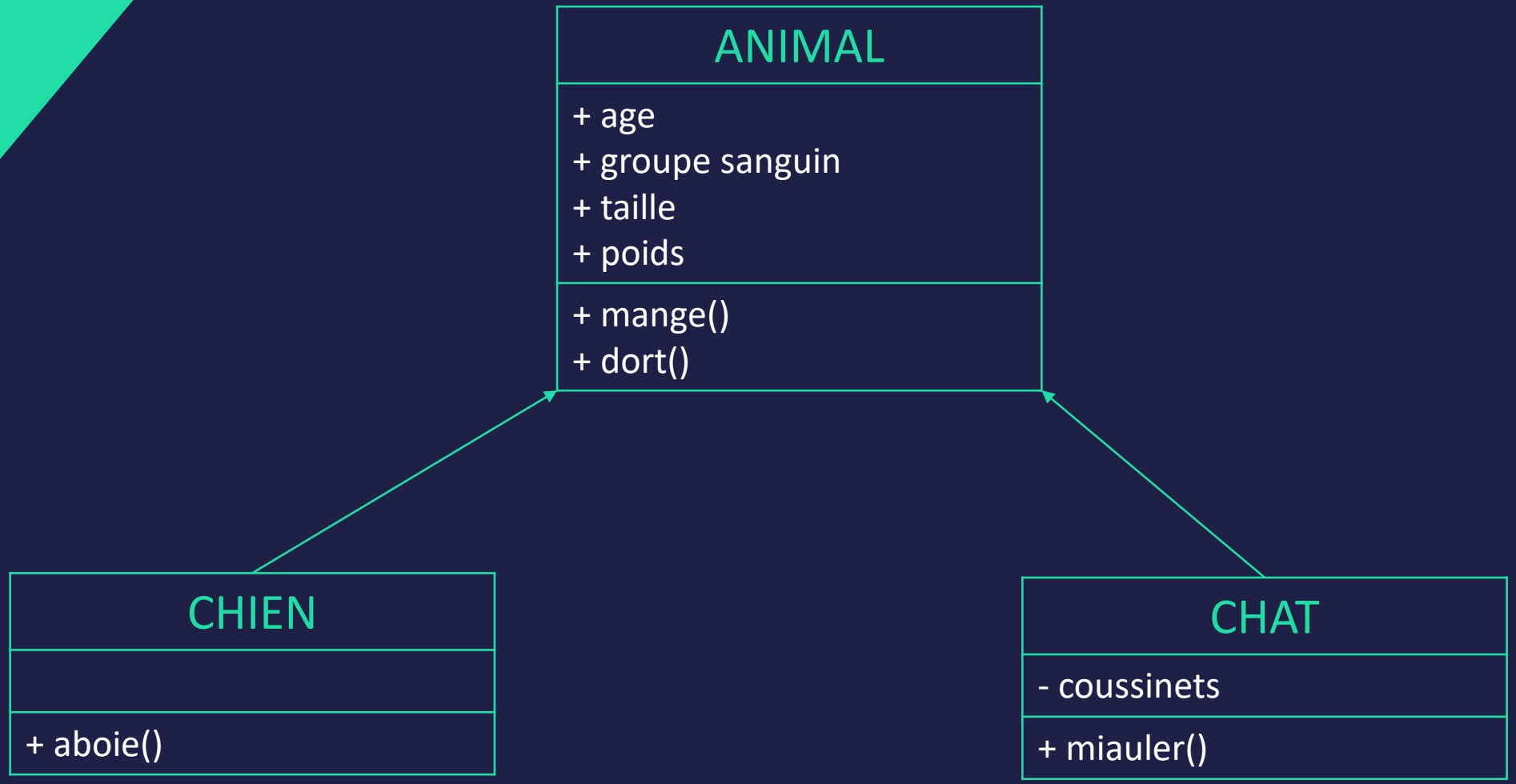
XEBIA NATIS

???
+ age - groupe sanguin # taille + poids
+ mange() + dort()

CHIEN
+ aboie()

CHAT
- coussinets
+ miauler()

XDNATIS


ANIMAL
+ age
+ groupe sanguin
+ taille
+ poids
+ mange()
+ dort()
CHIEN
+ aboie()
CHAT
- coussinets
+ miauler()

XWIKI NATIS

L’héritage permet d’organiser les classes en groupe plus spécifiques qui ont des propriétés et des méthodes qui leur sont propres

XWNATIS

L’héritage se fait au niveau de la définition de la classe

```python
from abc import ABC, abstractmethod

class Vehicule(ABC):
    @abstractmethod
    def transporter(self) -> None:
        pass

class Voiture(Vehicule):
    def transporter(self) -> None:
        print('Je roule')

class Avion(Vehicule):
    def transporter(self) -> None:
        print('Je vole')
```

XWIKI NATIS

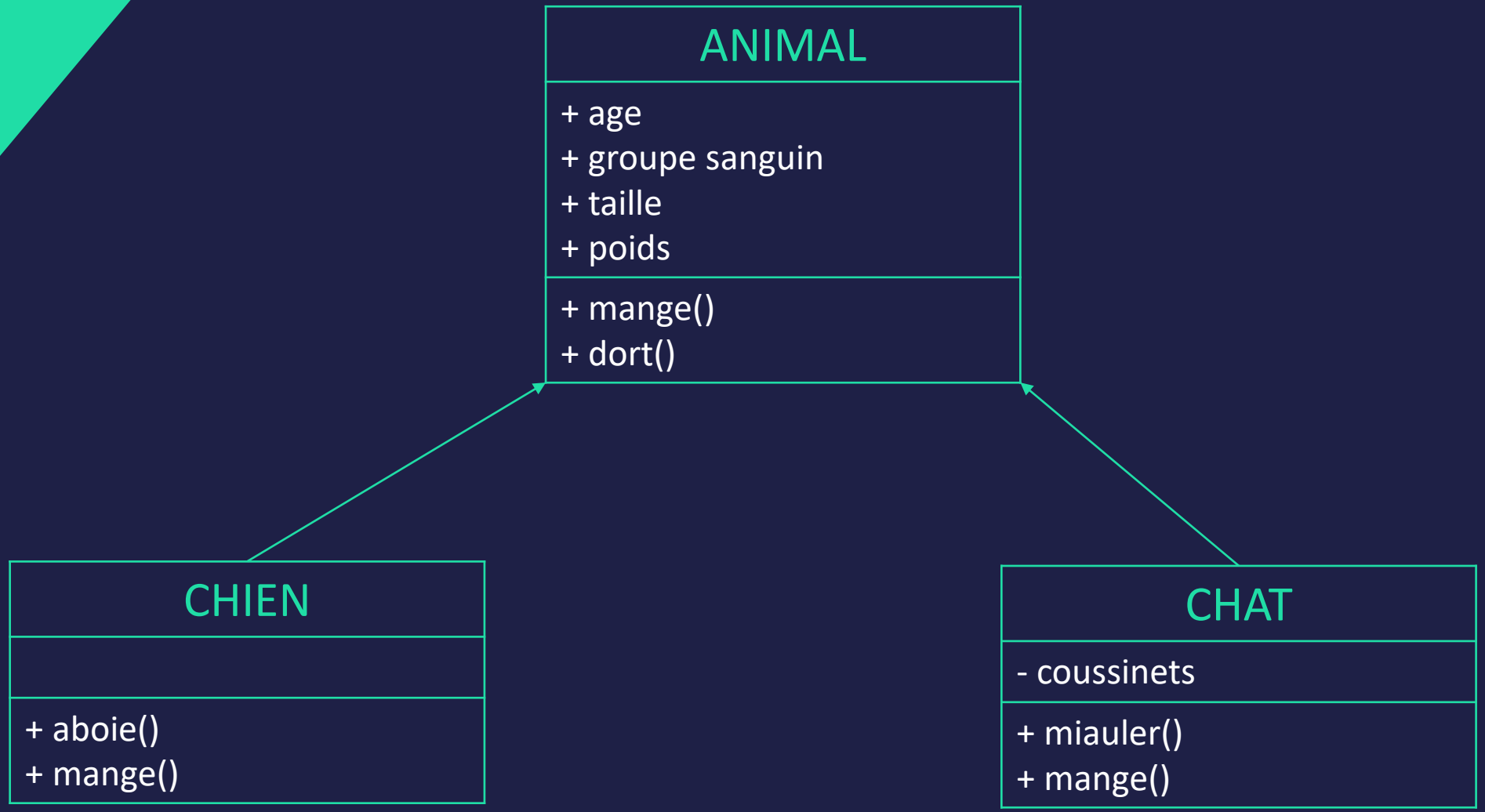
XEBIA NATIS

1. Les classes
2. L’intanciation
3. L’abstraction
4. L’héritage
5. Le polymorphisme
6. L’encapsulation

XEBIA NATIS

LE POLYMORPHISME

XEBIA NATIS


ANIMAL
+ age
+ groupe sanguin
+ taille
+ poids
+ mange()
+ dort()
CHIEN
+ aboie()
+ mange()
CHAT
- coussinets
+ miauler()
+ mange()

XWIKI NATIS

Le polymorphisme (de méthode) permet d'utiliser le même verbe pour décrire 2 méthodes différentes

XEBIA NATIS

```python
from abc import ABC, abstractmethod

class Vehicule(ABC):
    @abstractmethod
    def transporter(self) -> None:
        pass

class Voiture(Vehicule):
    def transporter(self) -> None:
        print('Je roule')

class Avion(Vehicule):
    def transporter(self) -> None:
        print('Je vole')
```

XWIKI SAS

XEBIA NATIS

1. Les classes
2. L’intanciation
3. L’abstraction
4. L’héritage
5. Le polymorphisme
6. L’encapsulation

XEBIA NATIS

L'ENCAPSULATION

XONATIS

XONATIS

XONATIS

XWIKI NATIS

L’encapsulation permet de définir une propriété ou une méthode interne

Public : accessible partout
Privé : interne à l‘objet et non hérité
Protégé : interne à l‘objet et hérité

XEBIANATIS

L'encapsulation n'existe pas en soi en Python mais est mimiqué par le nombre de underscores qui préfixe un membre

```python
class Voiture:
    def __init__(self) -> None:
        self.roues = 'Michelin' # public
        self._volant = 'Bosch' # protected
        self.__electronic = 'Panasonic' # private
```

XEBIA NATIS

4 PRINCIPES

ABSTRACTION

XWIKI NATIS

4 PRINCIPES

ABSTRACTION
HERITAGE

XKNATIS

4 PRINCIPES

ABSTRACTION
HERITAGE
POLYMORPHISME

XEBIA NATIS

4 PRINCIPES

ABSTRACTION
HERITAGE
POLYMORPHISME
ENCAPSULATION

XEBIA NATIS

4 PRINCIPES

ABSTRACTION
HERITAGE
POLYMORPHISME
ENCAPSULATION

XEBIA NATIS

4 PRINCIPES

ABSTRACTION
HERITAGE
POLYMORPHISME
ENCAPSULATION

XONATIS