## Plus loin avec les méthodes

#### Méthodes statiques

Vous pouvez souhaiter modifier le comportement d’une méthode reçue en héritage. Pour remplacer celle héritée, il suffit qu’elle soit accessible à la classe enfant : soit elle est privée mais présente dans la même unité, soit elle est d’une visibilité supérieure et accessible y compris dans une autre unité.

Par exemple, en ce qui concerne la méthode *Manger* définie dans l’ancêtre *TAnimal*, vous estimerez à juste titre qu’elle a besoin d’être adaptée au régime d’un carnivore. Afin de la redéfinir, il suffirait de l’inclure à nouveau dans l’interface puis de coder son nouveau comportement :

TChien = class(TAnimal)

strict private

fBatard : Boolean ;

procedure SetBatard;

public

procedure Manger; // la méthode est redéfinie

procedure Aboyer;

procedure RemuerDeLaQueue;

property Batard: Boolean read fBatard write SetBatard;

end;

#### Méthodes virtuelles

#### Méthodes abstraites

#### Méthodes de classes

## Les propriétés

Si une propriété ressemble à une variable, elle n’en est pas une dans la mesure où l’affectation d’une valeur à une propriété peut déclencher l’exécution d’une méthode. Cette dernière, en plus d’accéder au champ visé, pourra effectuer des traitements particuliers nécessaires au programme.

## La POO en action

### La variable Self

### Propriétaire et parent

Lorsque vous utilisez des composants dont le propriétaire est défini (ce qui est le cas dès que vous vous servez des fiches et que vous y déposez les composants nécessaires à votre application), c’est ce propriétaire qui est chargé d’allouer et de libérer la mémoire de manière transparente. Vous n’avez par conséquent pas à vous en occuper.

var

MonAnimal : TAnimal ;

begin

MonAnimal  := TAnimal.Create ; // on crée la liste de chaînes

try // on protège le code de manipulation pour être sûr de préserver les ressources

// ici le traitement voulu…

MonAnimal.Dormir ;

finally

// en interne, la méthode *Free* appelle le destructeur *Destroy*

MonAnimal .Free ; // les ressources seront toujours libérées

end ;

end ;