# Héritage et constructeurs, complément

Les constructeurs d'une classe dérivée (fille), avant d'exécuter leurs initialisations, font forcément appel à un constructeur de la classe de base.

Ainsi, si dans la classe de base *Rond* nous avons uniquement :

```
public Rond ()
{
}
```

Alors dans la classe dérivée *Cylindre,* si aucun constructeur n'est défini, c'est comme si nous avions écrit ceci :

```
public Cylindre () : base ()
{
}
```

Un constructeur par défaut est créé par le système et le constructeur par défaut de la classe parente, base(), est appelé.

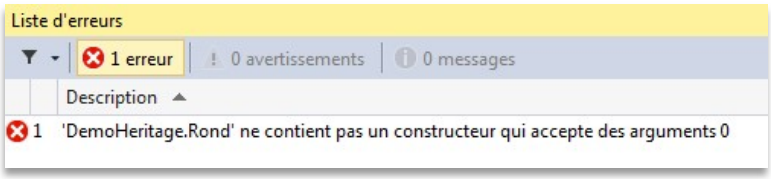
Tout va bien ça compile.

Par contre, si dans la classe de base *Rond* nous avons **uniquement** :

```
public Rond (double rayon)
{
}
```

Dans la classe dérivée *Cylindre,* si aucun constructeur n'est défini, c'est toujours comme si nous avions écrit ceci :

```
public Cylindre () : base () → Erreur de compilation !
{
}
```




C'est normal car dans ce cas `base ()` veut dire qu'il faut appeler le constructeur `Rond ()` qui n'existe pas !

Le bon usage voudrait que pour un constructeur dans la classe de base nous ayons « l'équivalent » dans la classe dérivée :

```
public Rond ()
…
public Cylindre () : base ()
{

}

public Rond (double rayon)
…
public Cylindre (double rayon, double hauteur) : base (rayon)
{
  this.Hauteur = hauteur ; // On s'occupe uniquement de la nouveauté
}
```