

# Programação Orientada a Objetos

## Aula 06 — Herança

Hugo Marcondes

Departamento Acadêmico de Eletrônica  
DAELN

[hugo.marcondes@ifsc.edu.br](mailto:hugo.marcondes@ifsc.edu.br)



Notes

---

---

---

---

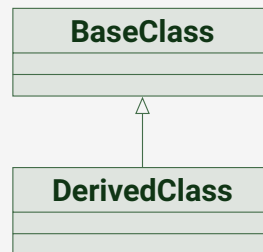
---

---

---

## Herança — C++

- Herança permite a definição de uma classe em termos de outra classe
  - Classe Base
  - Classe Derivada



```
class DerivedClass: access-specifier BaseClass
```

Notes

---

---

---

---

---

---

---



## Exemplo

```
1 // Base class
2 class Shape {
3     public:
4         void setWidth(int w) { width = w; }
5         void setHeight(int h) { height = h; }
6     protected:
7         int width;
8         int height;
9 };
10
11 // Derived class
12 class Rectangle: public Shape
13 {
14     public:
15         int getArea() { return (width * height); }
16 };
17
```



## Notes

---

---

---

---

---

---

---

## Exemplo

```
1 int main(void) {
2     Rectangle Rect;
3
4     Rect.setWidth(5);
5     Rect.setHeight(7);
6
7     // Print the area of the object.
8     cout << "Total area: " << Rect.getArea() << endl;
9
10    return 0;
11 }
12
```



## Notes

---

---

---

---

---

---

---

## Herança

- Uma classe derivada herda todos os atributos e métodos da classe base, com as seguintes exceções:
- Construtores, Destrutores e Construtores de cópia da classe Base
- Operadores sobrecarregados da classe Base
- Funções “amigas” (friend functions) da classe Base



## Notes

---

---

---

---

---

---

## Controle de Acesso

- Os atributos e métodos de uma classe podem ser qualificadas de acordo com a sua acessibilidade.
- Os qualificadores de acesso em C++ podem ser sumarizados de acordo com a tabela abaixo:

Acesso	public	protected	private
Interno	✓	✓	✓
Classe Derivada	✓	✓	✗
Externo	✓	✗	✗



## Notes

---

---

---

---

---

---

## Controle de Acesso na Herança

- Em C++ as heranças também podem ser qualificadas com qualificadores de acesso
- Determinam a visibilidade dos métodos herdados pela classe Base, na classe Derivada
  - Herança Pública (**public**)
    - Não alteram os qualificadores dos métodos da classe Base, na classe Derivada.
  - Herança Protegida (**protected**)
    - Os métodos público e protegidos da classe Base, se tornam métodos protegidos na classe Derivada
  - Herança Privada (**private**)
    - Os métodos públicos e protegidos da classe Base, se tornam métodos privados na classe Derivada



Notes

---

---

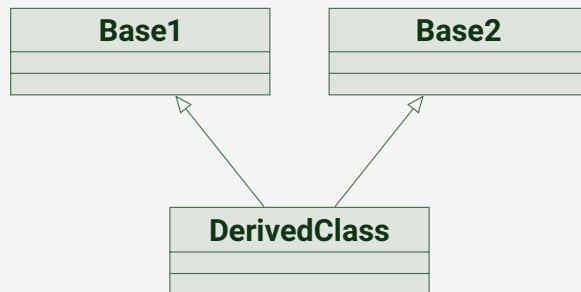
---

---

---

---

## Herança Múltipla – C++



```
class DerivedClass: access Base1, access Base2, ...
```



Notes

---

---

---

---

---

---

## Exemplo

```
1 // Base class Shape
2 class Shape {
3     public:
4         void setWidth(int w) { width = w; }
5         void setHeight(int h) { height = h; }
6     protected:
7         int width;
8         int height;
9 };
10 // Base class PaintCost
11 class PaintCost {
12     public:
13         int getCost(int area) { return area * 70; }
14 };
15 // Derived class
16 class Rectangle: public Shape, public PaintCost {
17     public:
18         int getArea() { return (width * height); }
19 };
```



## Notes

---

---

---

---

---

---

---

## Exemplo

```
1 int main(void) {
2     Rectangle Rect;
3     int area;
4
5     Rect.setWidth(5);
6     Rect.setHeight(7);
7
8     area = Rect.getArea();
9     // Print the area of the object.
10    cout << "Total area: " << Rect.getArea() << endl;
11
12    // Print the total cost of painting
13    cout << "Total paint cost: $";
14    cout << Rect.getCost(area) << endl;
15
16    return 0;
17 }
```



## Notes

---

---

---

---

---

---

---

That's all folks!



**INSTITUTO  
FEDERAL**  
Santa Catarina  

---

Câmpus  
Florianópolis



Notes

---

---

---

---

---

---

---

Notes

---

---

---

---

---

---

---