iOS

Criando Aplicações para iPhone e iPad

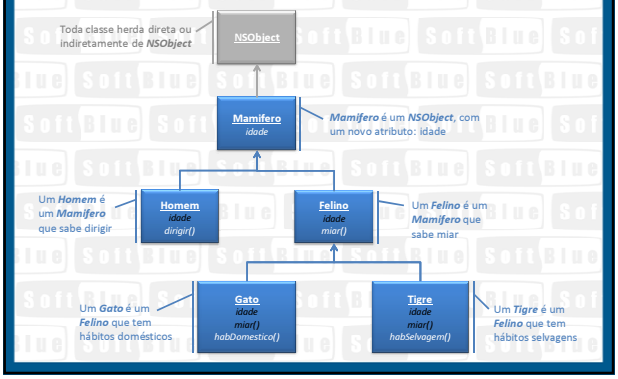
Herança

SoftBlue
www.softblue.com.br

Herança

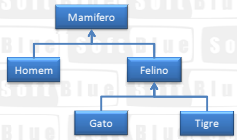
- Permite que uma classe filha (subclasse) herde elementos não privados (***private***) de sua classe pai, também conhecida como superclasse
- Benefícios:
  - Reaproveitamento de código
  - Organização do código
  - Facilidade de manutenção
- Em algumas linguagens é possível utilizar herança múltipla
- Na omissão, herda-se da classe ***NSObject***

Hierarquia de classes



Herdando uma classe

- Operador **:**




```
@interface Mamifero {

    int idade;
    int peso;

}
- (void) andar {
    // Método que anda
}
@end
```

```
@interface Homem {

    int idade;
    int peso;
    int cpf;

}
- (void) andar {
    // Método que anda
}

- (void) dirigir {
    // Método que dirige
}
@end
```

```
@interface Homem : Mamifero {

    int idade;
    int peso;
    int cpf;

}
public function andar() {
    // Método que anda
}
-
 (void) dirigir {
    // Método que dirige
}
@end
```

Construtores hierárquicos

- Operador ***super***

```
@interface Mamifero {

    int idade;
    int peso;

}

- (id) initWithIdade: (int) i peso: (int) p {
    // Validações de idade e peso
    $self->idade = idade;
    $self->peso = peso;
    return self;
}

@end
```

```
@interface Homem : Mamifero {

    int cpf;

}


- (id) initWithIdade: (int) i peso: (int) p cpf: (int) c {

    [super initWithIdade: i peso: p];

    // Validações de cpf
    $self->cpf = cpf;
    return self;
}

@end
```

Sobrescrita de métodos

- Métodos herdados podem ser sobrescritos
- Subclasse se comporta de maneira diferente para a mesma invocação do método
- Métodos sobrescritos substituem para a classe em questão os métodos da superclasse
- Superclasse continua com seu comportamento habitual, não sofrendo alterações
- Na subclasse não é necessário criar a assinatura do método (arquivo .h)

Sobrescrevendo métodos

```
@interface Mamifero {

}

- (int) andar {
    // Método que anda de mamífero
    return 1;
}
@end
```

```
@interface Homem : Mamifero {

}

- (int) andar {
    // Método que anda (sobreescrito)
    return 2;
}
@end
```

```
Mamifero *m = [[Mamifero alloc] init];
[m andar]; // andar(mamifero): retorna 1

Homem*h = [[Homem alloc] init];
[h andar]; // andar(homem): retorna 2
```

Restringindo acesso

- Operador ***protected***
- Restringe o acesso aos atributos e aos métodos definidos para somente as subclasses

```
@interface Mamifero {

    @protected
    int variavel;
}

@end
```

```
@interface Homem : Mamifero {

}
- (int) algumMetodo {
    return [super variavel];
}
@end
```

```
Mamifero *m = [[Mamifero alloc] init];
[m variavel];                // erro

Homem*h = [[Homem alloc] init];
[h algumMetodo];         // retorna o valor de variavel
```

Aulas práticas e manuais on-line

Assista agora às aulas práticas.

Clique aqui para visualizar as aulas práticas disponíveis