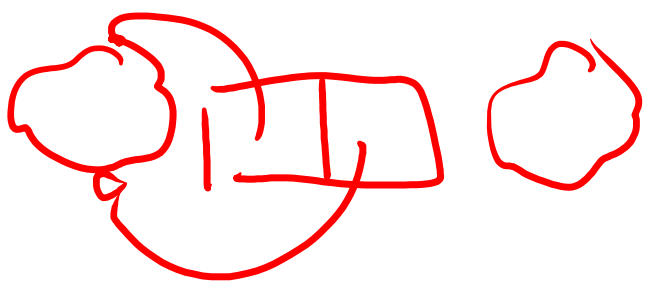
Computação II – Orientação a Objetos

Fabio Mascarenhas - 2016.2

http://www.dcc.ufrj.br/~fabiom/java

Identidade de objetos

- Java também tem as operações de comparação `==` (igual a) e `!=` (diferente de)
- Para números e booleanos elas dizem se dois números ou booleanos são iguais ou não
- Mas, para objetos (incluindo strings e vetores), essas operações dizem se os dois objetos que estamos comparando são *a mesma instância* ou não
- Todo objeto possui uma identidade; quando criamos um objeto com `new`, a identidade dele é diferente de todos os outros objetos, mesmo objetos da mesma classe, com o conteúdo dos campos idêntico



Métodos

- Um *método* é uma operação de um objeto

- Sintaticamente, um método se parece com uma função: é declarado dentro de uma classe (mas sem o palavra-chave `static`), possui uma lista de parâmetros e um bloco de comandos, e pode retornar valores

- Mas, para chamar um método, precisamos dizer qual objeto vai *receber* a chamada: `<obj_rec>.método(<arg1>, ..., <argn>)`

  - `"ab".equals("a" + "b"), scan.nextInt()`

- Dentro de um método, o variável this é o objeto que está recebendo a chamada, e podemos ter acesso aos campos e métodos desse objeto

Comparando por conteúdo

- Comparar strings com `==` não funciona, às vezes duas strings com o mesmo conteúdo acabam apontando para a mesma instância de `String`, às vezes não

- Se queremos comparar objetos por *conteúdo* ao invés de por identidade, precisamos usar o método `equals`

  - `"ab".equals("a" + "b") == true`

- Todo objeto tem um método equals, mas nem sempre ele funciona

  - `(new int[] { 1 }).equals(new int[] { 1 }) == false`

Frameworks

- É bastante comum que uma aplicação OO seja construída para usar um *framework* (arcabouço)
- Um framework é como uma máquina com algumas peças faltando, que serão fornecidas pela aplicação
- Essas “peças faltando” são instâncias de classes cuja estrutura é ditada pelas necessidades do framework
- Os objetos do framework interagem com os objetos da aplicação, que por sua vez interagem de volta com objetos do framework, para requisitar serviços

Um framework simples para jogos

- Vamos usar um framework bem simples para construir jogos 2D
- O framework fornece uma *tela* para desenhar figuras geométricas simples, além de texto
- A cada “tique” do relógio interno do framework, ele fornece uma tela em branco
- Ele também avisa a aplicação de *eventos* que acontecem: teclas pressionadas, e a própria passagem do tempo
- O jogo fornece ao framework algumas informações como seu título e as dimensões de sua tela

Comunicação framework vs. jogo

- Toda a comunicação do framework com a jogo se dá através de métodos

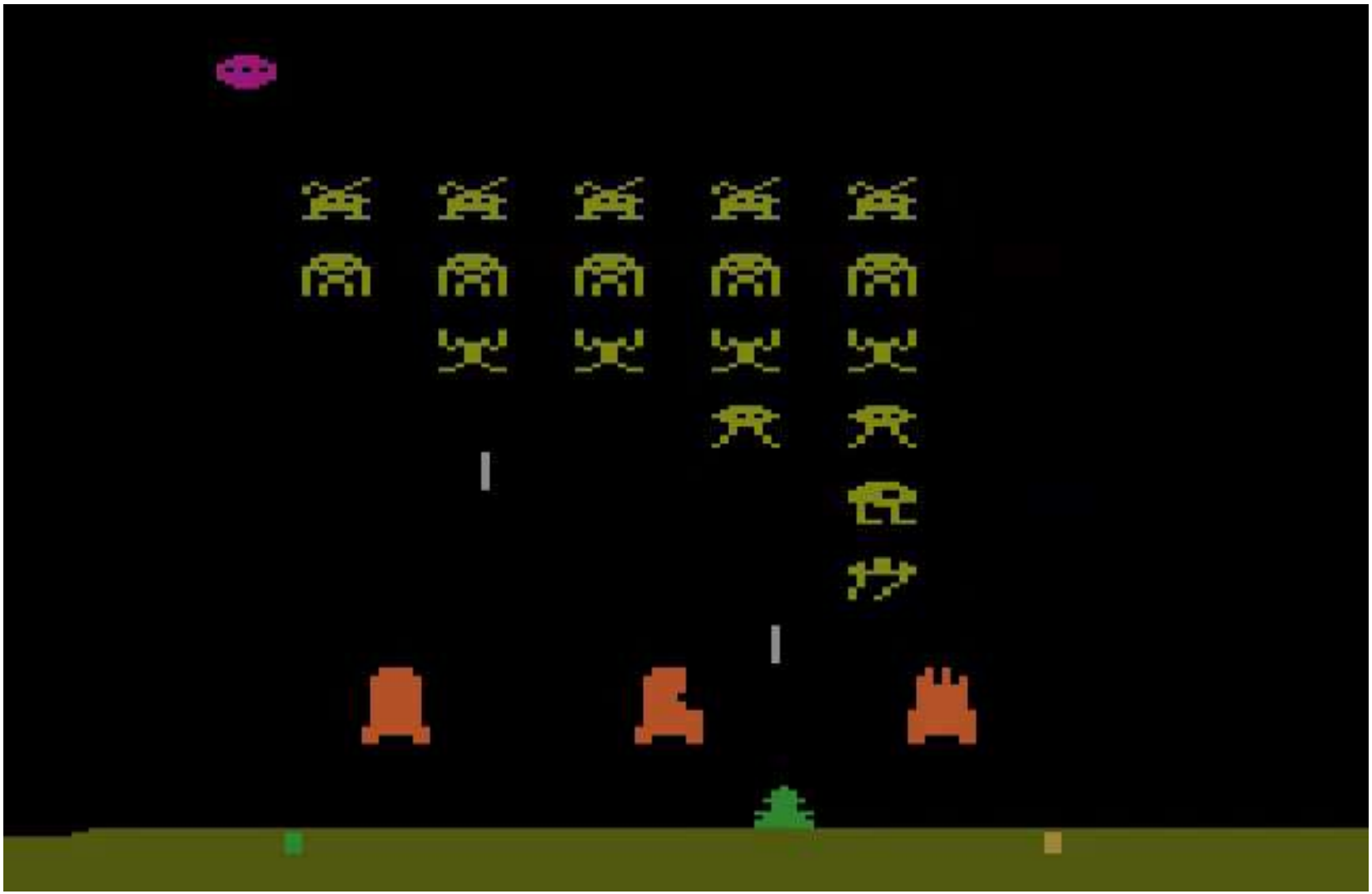
- O jogo (uma classe `Jogo`) define seis métodos


```
String getTitulo()
int getAltura()
int getLargura()
void tecla(String tecla)
void tique(Set<String> teclas, double dt)
void desenhar(Tela tela)
```

- A classe `Tela` define outros seis

```
public void triangulo(double x1, double y1,
        double x2, double y2, double x3, double y3, Cor cor)
public void circulo(double cx, double cy, int raio, Cor cor)
public void quadrado(double x, double y, int lado, Cor cor)
public void retangulo(double x, double y, int largura, int altura, Cor cor)
public void texto(String texto, double x, double y, int tamanho, Cor cor)
public void imagem(String arquivo, int xa, int ya, int larg, int alt,
                   double dir, double x, double y)
```

Space Invaders



Componentes do Jogo

- Canhão
- Aliens
- Tiros
- Escudos
- Score e vidas
- Chão
- Nem todos vão precisar de classes próprias para representa-los!