

Programação Multimédia

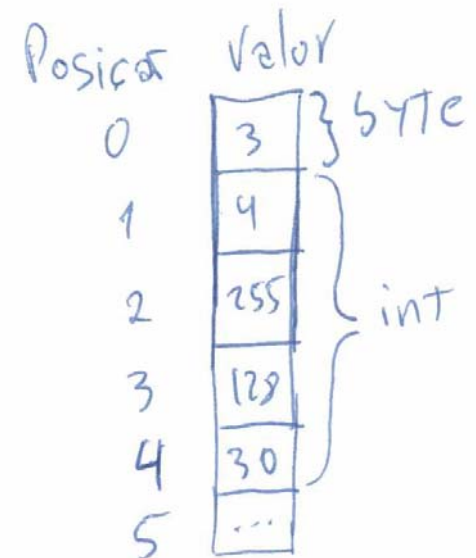
Variáveis

Memória do Programa (Variáveis)

- Permite-nos armazenar valores (dados)
- Os dados são guardados na memória RAM do computador
- As variáveis representam posições de memória que podem guardar um valor
- As variáveis têm um nome que utilizamos no programa
- Não precisamos de decorar a posição de memória

Memória

- A memória RAM é vista pelo processador como uma tabela
- Cada entrada da tabela pode armazenar um byte
- Cada entrada é referenciada pela sua posição (endereço)
- Um byte é um conjunto de 8 bits
 - Pode representar um número até 255 (em decimal)
- Cada entrada pode ser vista de forma isolada (byte) ou agrupada com outras posições



Memória

- É importante para o programador poder utilizar números maiores do que 255!!!
- As posições de memória podem ser agrupadas em conjuntos de bytes
- Os bytes podem ser estruturados de formas diferentes
 - valores inteiros, decimais, texto

Tipos de Dados

- Cada variável pode guardar valores de um determinado tipo:
 - int, long
 - float, double
 - boolean
 - char
- Tipos simples

Variáveis na Prática

- Num programa as posições de memória são representadas por nomes que o programador atribui
- Se precisamos de guardar uma idade, damos o nome “idade” à posição de memória
- O compilador “traduz” os nomes em posições de memória que o processador entende
- Para além do nome temos de indicar o tipo de variável
 - tipos diferentes têm tamanhos diferentes logo usam mais ou menos bytes contíguos na memória
 - o compilador tem de saber para poder alocar correctamente as posições de memória

Declaração de Variáveis (processing)

```
[tipo] nomeDaVariavel;
```

- Isto declara a variável apenas
- O Programa fica “a saber” que vamos utilizar uma posição de memória para guardar dados de um determinado tipo
- Exemplo:

```
int idade;
```
- Nota:
 - O ponto e vírgula “;” é importante!!!
 - É usado para terminar **todas** as instruções em Processing
 - Não o colocar é um erro sintáctico

Atribuição

- Quando a variável é criada, não tem nenhum valor guardado!
- Para guardarmos valores numa variável temos de *atribuir* esses valores:
`idade = 31;`
- A atribuição é feita com o sinal “igual”
 - Colocar na variável “idade” o valor 31
 - A partir daqui quando usamos a variável “idade”, estamos, de facto, a usar o valor 31
- Em qualquer altura do programa podemos atribuir um valor a uma variável

Inicialização

- Antes de usar uma variável é preciso colocar algo lá dentro
 - Inicializar a variável
- Porquê?
 - Qual o resultado do programa seguinte?

```
int x;
```

```
int y;
```

```
point(x, y);
```

- Experimentem!

Variáveis: Examples->Data->Variables

```
// Variables
// by REAS <http://reas.com>

// Variables are used for storing values.
// Changing the values of variables 'a' and 'b' significantly change the composition.

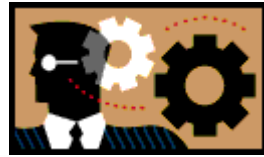
// Created 2 September 2002

size(200, 200);
background(0);
stroke(153);

int a = 20;
int b = 50;
int c = a*8;
int d = a*9;
int e = b-a;
int f = b*2;
int g = f+e;

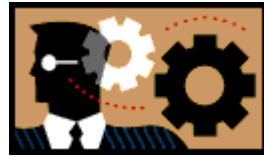
line(a, f, b, g);
line(b, e, b, g);
line(b, e, d, c);
line(a, e, d-e, c);
```

Visibilidade (scope)



- Podemos declarar variáveis dentro e fora dos métodos
- O que são métodos?
 - Para já, podemos pensar neles como conjuntos de instruções ao qual demos um nome
- Uma variável declarada dentro de um método só pode ser usada dentro desse método

Visibilidade (scope)



- Podemos declarar variáveis com o mesmo nome dentro e fora de um método ou em métodos diferentes:

```
void setup() {  
    int a;  
    a = 2;  
    mudaA();  
    print(a);  
}
```

```
void mudaA() {  
    int a;  
    a = 3;  
}
```

- Exemplo: Variable_Scope (Examples->Data->Variable_Scope)

Expressões

- A atribuição de valores a variáveis não tem de ser feita com valores simples (literais)
- Podemos atribuir a uma variável o resultado de uma expressão mais complicada:

```
int a;
```

```
int b;
```

```
a = 3*5;
```

```
b = a*35+22;
```

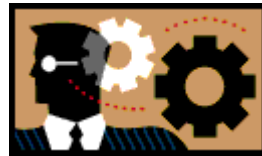
Operadores Aritméticos

- Adição: “+”
- Subtracção: “-”
- Divisão: “/”
- Multiplicação: “*”
- Resto da Divisão Inteira: “%”

Exercício

- Implementar programas para efectuar as seguintes operações
 - $32*32+10*10$
 - $x*x+y*y$ (atribua os valores que quiser a x e y)
 - $\text{jogosGanhos}*3+\text{jogosEmpatados}$
 - $\text{peso}/\text{altura}^2$

Atribuição com métodos



- Nota:
 - Métodos são conjuntos de instruções com nome que, nalguns casos, representam um valor
- Podemos também usar alguns métodos nas atribuições:

```
float a;  
a = 3*random(100);
```
- `random()`
 - Procurar na referência
- Podemos usar métodos que devolvem valores
 - Aula sobre funções

Exemplo: Animar um ponto

```
int x = 0;
int y = 0;

void setup() {
  size(200, 200);
  framerate(5);
}

void draw() {
  background(0); // limpar o fundo (pintar a preto)

  stroke(255);   // usar a cor branca para pintar

  point(x, y);  // desenhar um ponto branco na posicao (x, y)

  // actualizar a posicao (x, y)
  x = x + 1;
  y = y + 1;
}
```

Exemplo: Animar um ponto

- Modificar a expressão de actualização da posição:

```
x = x + 1 ;
```

```
y = y + 1 ;
```

para

```
x = x + 2 ;
```

```
y = y + 5 ;
```

Projecto Semanal

1. <http://teaching.jorgecardoso.org/pm200620071sem/programa.php>