CENIN

Curso Java Básico

Uma introdução prática usando BlueJ

Objetos e Classes

Fundamentos da Orientação a Objetos

Principais conceitos a serem abordados

- Objeto
- Classe
- Método
- Parâmetro
- Campo
- Tipo de dados

Objetos e classes

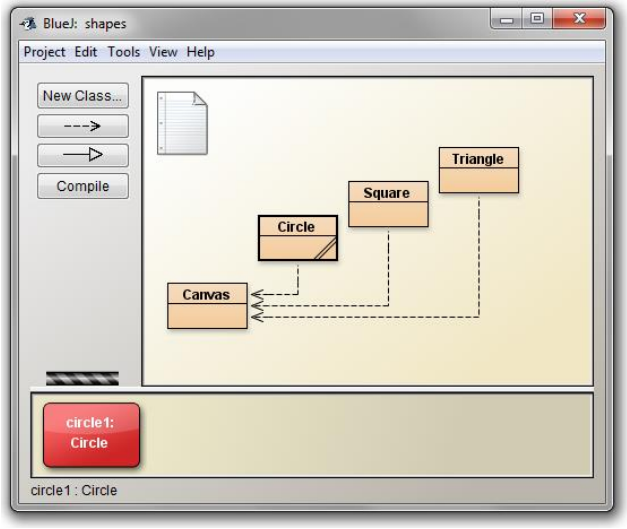
- Objetos
  - Representam ‘coisas’ do mundo real ou do domínio de algum problema (exemplo: “o carro vermelho ali no estacionamento”).
- Classes
  - Representam todos os tipos de objetos (exemplo: “carro”).

Objetos e Classes

- Objeto = instância de uma classe
  - Objetos são criados a partir de classes
  - Por convenção:
    - nomes de classes iniciam por *maiúsculas*
    - nomes de objetos iniciam por *minúsculas*

Exercício

- **Criando objetos**
  - Inicie o **BlueJ** e abra o projeto *shapes*
  - A partir da classe *Circle*, crie um objeto (aceite o nome *circle1*)

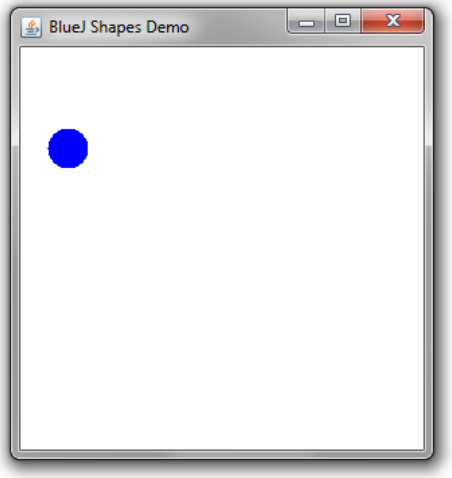


Chamando métodos

- Objetos realizam operações através de métodos
- Métodos são chamados (ou invocados)

Exercício

- **Chamando métodos**
  - A partir do objeto *circle1*, chame o método *makeVisible*
  - Chame também os métodos *moveRight* e *moveDown*



Passando parâmetros

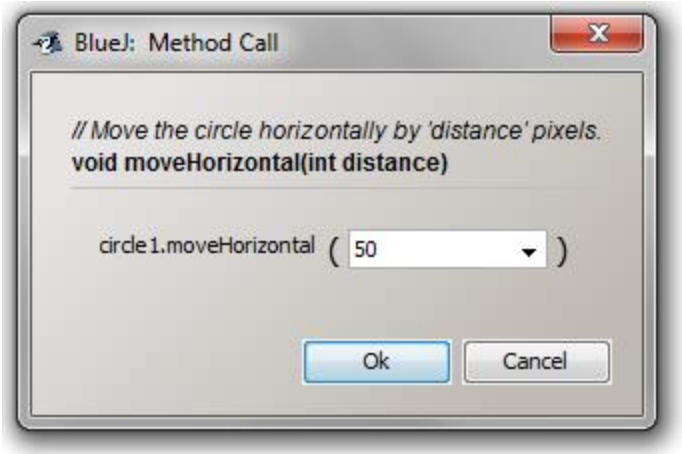
- Métodos podem ter parâmetros para fornecer informações adicionais para realizar uma tarefa
- No cabeçalho do método, os parâmetros são listados após o nome do método entre parênteses e separados por vírgulas
- Cada parâmetro tem nome e tipo de dados

Tipos de dados

- O tipo de dado define o conjunto de valores que uma variável pode assumir:
  - O tipo *int* significa um número inteiro
  - O tipo *String* significa uma seção de texto (strings são informadas entre aspas duplas, como “red”)
  - Outros tipos de dados serão vistos depois

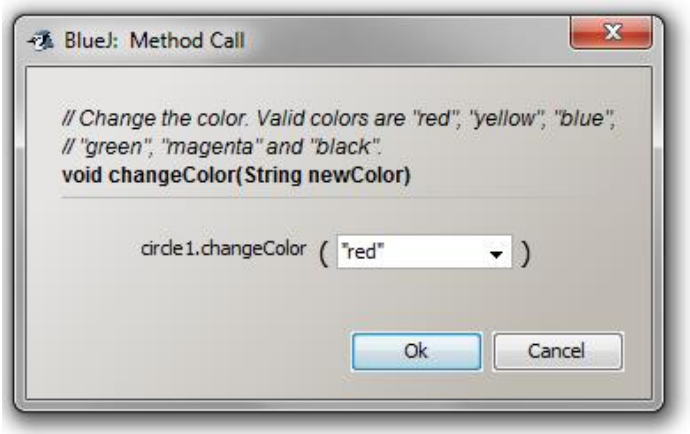
Exercício

- **Passando parâmetros**
  - Chame o método *moveHorizontal*, informando o valor *50* para *distance*
  - Chame os métodos *moveVertical*, *slowMoveVertical* e *changeSize*



Exercício

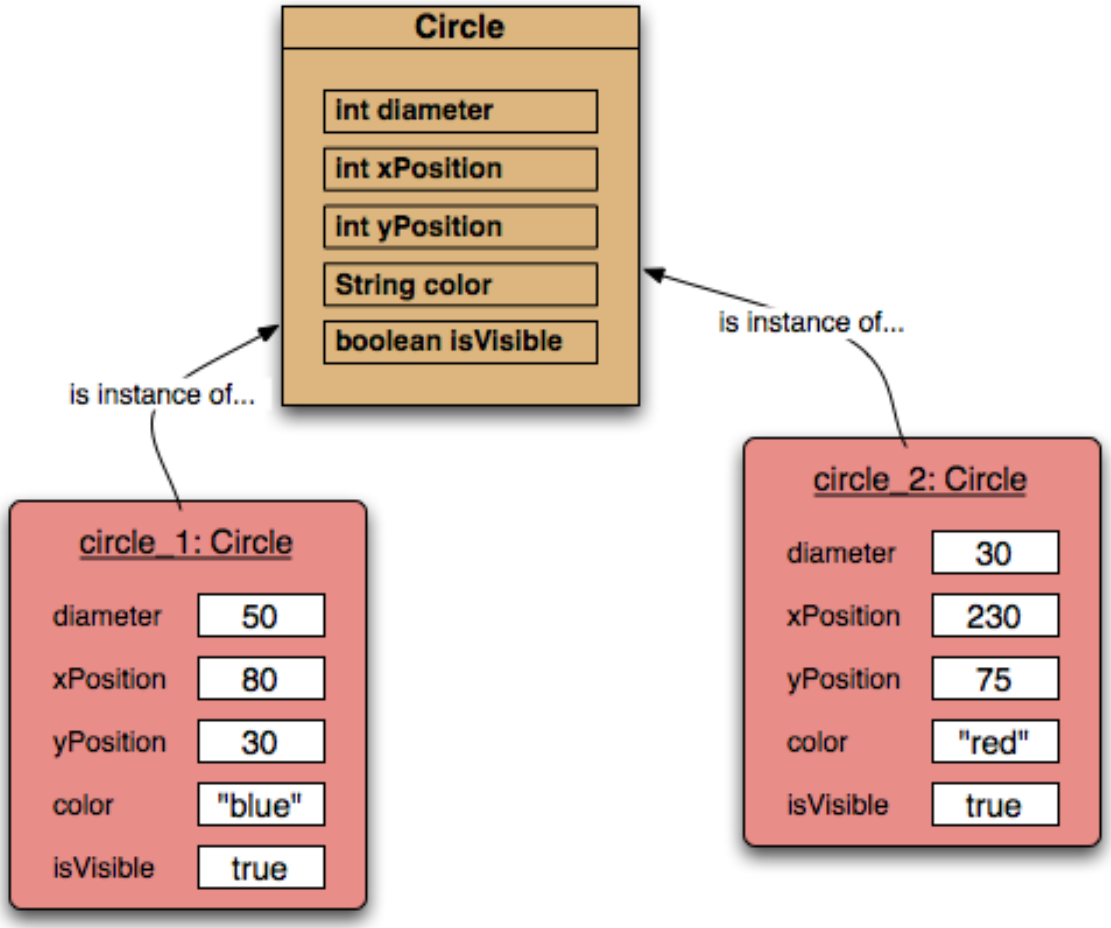
- **Passando parâmetros**
  - Chame o método *changeColor* e informe *"red"*
  - Chame o método *changeColor* e informe *red* e *"orange"*



Múltiplas instâncias

- Várias instâncias podem ser criados a partir de uma classe
- Todas as instâncias da mesma classe:
  - terão os mesmos atributos
  - executarão as mesmas operações

Dois objetos circle



Exercício

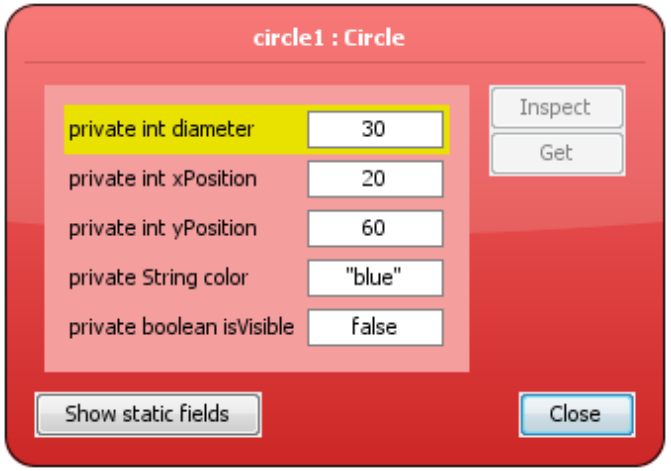
- **Criando objetos**
  - Crie outro círculo, um quadrado e um triângulo a partir das classes apropriadas
  - Chame o método *makeVisible* nos objetos criados

Estado

- O conjunto dos valores de todos os atributos de um objeto é chamado de estado do objeto

Estado

- No BlueJ, o estado de um objeto pode ser inspecionado selecionando a função *Inspect* a partir do menu pop-up do objeto



Exercício

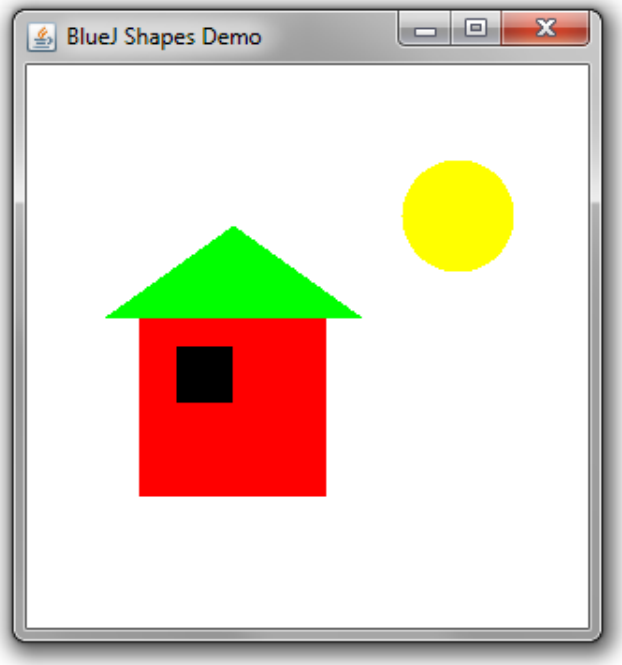
- **Estado**
  - Verifique o estado dos objetos na sua bancada de objetos
  - Altere o estado de algum objeto enquanto a janela do inspetor de objetos estiver aberta

Interação entre objetos

- Objetos podem criar outros objetos e chamar seus métodos
- Durante a execução de um programa orientado a objetos normal, diversos objetos são criados e interagem para executar as tarefas do programa

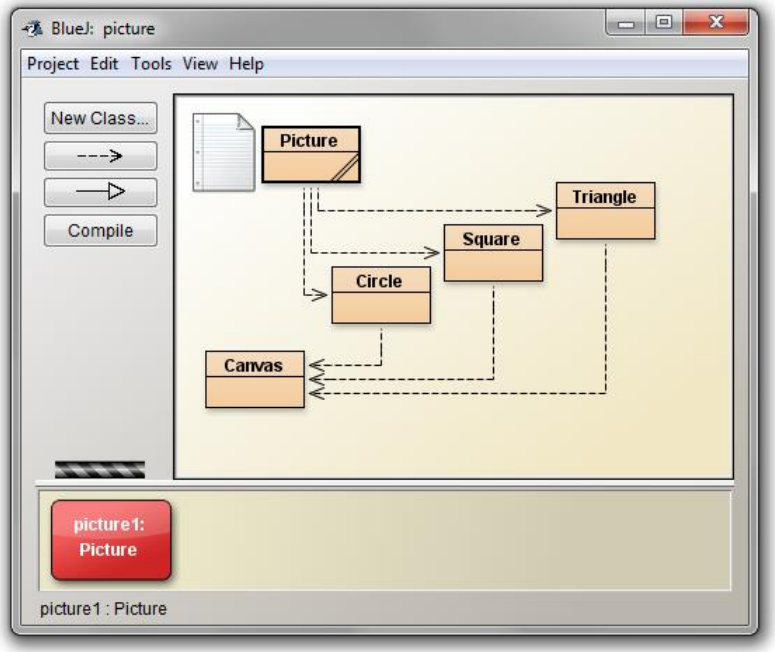
Exercício

- **Interação entre objetos**
  - Utilize objetos do projeto *shapes* para criar uma casa e um sol como nesta figura.



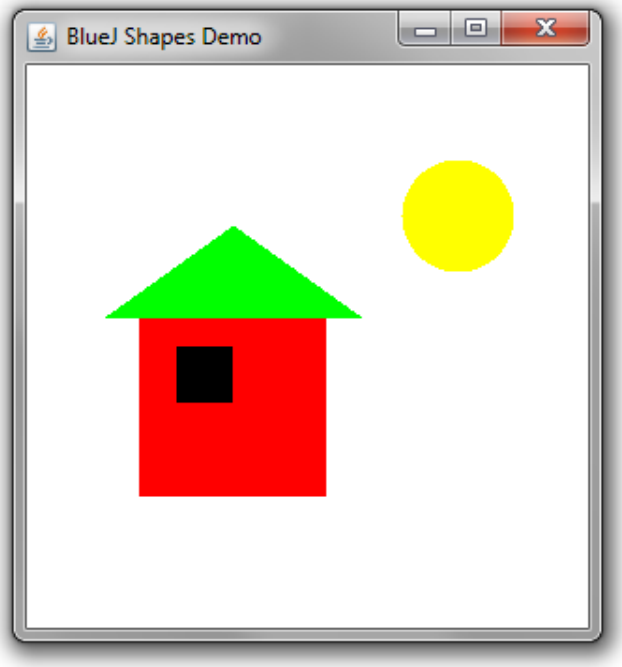
Exercício

- **Interação entre objetos**
  - Abra o projeto *picture* e crie uma instância da classe *Picture*.



Exercício

- **Interação entre objetos**
  - Invoque o método *draw* da instância de *Picture*.
  - Como você acha que a classe *Picture* desenha a figura ?



Código-fonte

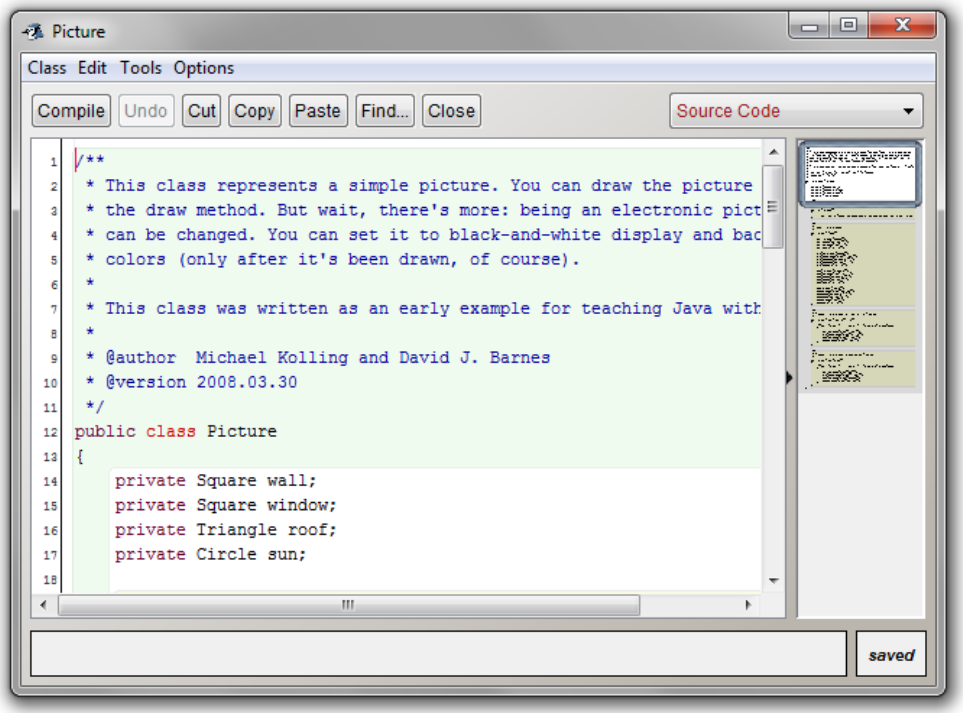
- Cada classe tem um código-fonte associado
- O código-fonte de uma classe determina a estrutura e o comportamento dos objetos desta classe

Código-fonte

- No BlueJ, o código-fonte de uma classe pode ser visualizado selecionando a função *Open Editor* a partir do menu pop-up da classe ou com um duplo clique no seu ícone
- Quando o código-fonte de uma classe é alterado, é necessário sua compilação

Exercício

- **Código-fonte**
  - Edite o código-fonte da classe *Picture*.



Exercício

- **Código-fonte**
  - Localize no código-fonte da classe *Picture* a parte que desenha a figura e altere-a de modo que o sol seja vermelho ao invés de amarelo. Desfaça a alteração.
  - Crie um segundo sol na figura, adicionando o campo *sun2* e as instruções apropriadas no método *draw*. Desfaça a alteração.

Exercício

- **Código-fonte**
  - Crie um pôr-do-sol para a classe *Picture* em seu método *draw* chamando um método apropriado da classe *Circle*.
  - Crie um pôr-do-sol para a classe *Picture* criando um método de nome *sunset*, de modo que possamos desenhar a figura e depois ver o pôr-do-sol.

Valores de retorno

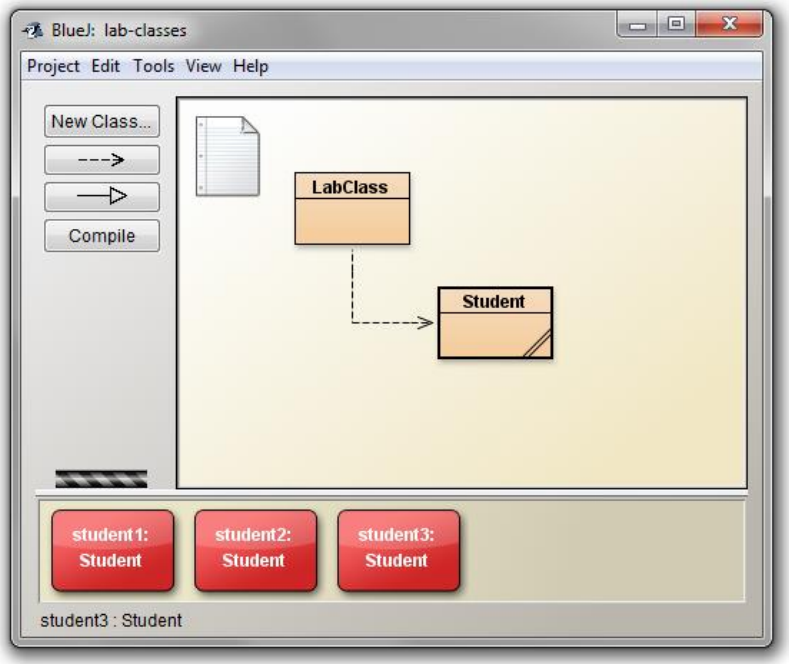
- Métodos podem retornar informações como resultado de sua execução
- O cabeçalho de um método informa o tipo do valor de retorno antes de seu nome
- O valor de retorno *void* significa que o método não retorna nenhum resultado

Valores de retorno

- Métodos podem retornar informações como resultado de sua execução
- O cabeçalho de um método informa o tipo do valor de retorno antes de seu nome
- O valor de retorno *void* significa que o método não retorna nenhum resultado

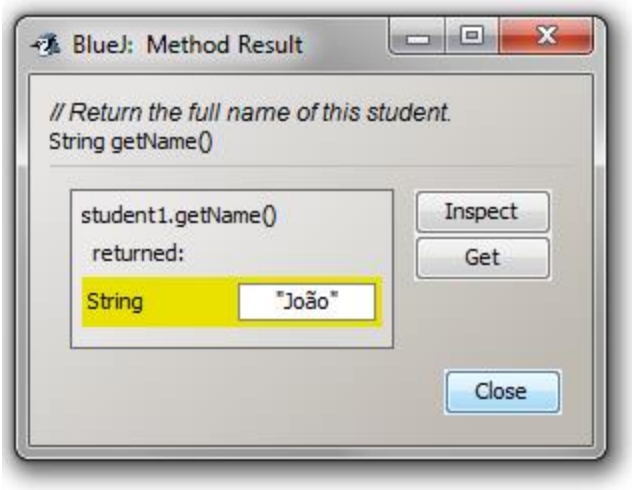
Exercício

- **Valores de retorno**
  - Abra o projeto *lab-classes* e crie algumas instâncias da classe *Student*.



Exercício

- **Valores de retorno**
  - Chame o método *getName* em um objeto da classe *Student* (observe o tipo do valor de retorno).
  - Inspecione o valor de retorno.



Objetos como parâmetros

- Objetos podem ser passados como parâmetros aos métodos de outros objetos.
- No caso de um método esperar um objeto como parâmetro, o nome da classe do objeto esperado é especificado como tipo de dado do parâmetro.

Exercício

- **Valores de retorno**
  - Crie um objeto da classe *LabClass*.
  - Chame os métodos apropriados para configurar instrutor, sala e dia-hora.
  - Chame o método *enrollStudent* para inscrever os alunos (com o cursor no campo de entrada do diálogo, clique no objeto aluno).
  - Chame o método *numberOfStudents*.
  - Chame o método *printList*.

Revisão (1)

- As classes representam o conceito geral de uma coisa e os objetos representam instâncias concretas de uma classe.
- Objetos são criados com base em definições de classe escritas em uma linguagem de programação.

Revisão (2)

- Os objetos tem métodos que realizam operações. Eles podem ter parâmetros e retorno, os quais têm tipo de dado.
- Os objetos armazenam dados em campos que também têm tipo de dado. O conjunto dos valores dos campos do objeto em um momento é chamado estado do objeto.

Contatos

Câmara dos Deputados

CENIN - Centro de Informática

Carlos Renato S. Ramos

carlosrenato.ramos@camara.gov.br