Programa do Minicurso

1 Identificação

Disciplina: Minicurso de Java Moderno

Carga Horária: 20 horas-aula Teóricas: 10 Práticas: 10

Período: Abril a Junho de 2017.

2 Cursos (público alvo)

Ciências da Computação;

Sistemas de Informação.

3 Requisitos

Nenhum.

4 Ementa

Aritmética em programação; Estruturas Condicionais e de Seleção; Estruturas de Repetição; Métodos, Procedimentos e Funções; Encapsulamento de atributos; Interfaces; Tipos definidos por usuário; Classes abstratas; Padrões de Projeto; Diagramas UML (*Unified Modeling Language*): Classes e Casos de Uso; Componentes básicos AWT (*Abstract Window Toolkit*); Padrão Observador-Observável; Padrão MVC; *Bitmaps*; *Double-Buffering*; *Page-Flipping*; *Callback*; Reuso; Boas Práticas de Programação (padrões de nomenclatura/casing, etc.); Funções *lambda*; *Cache friendliness*; *Generics*; Classe Anônima; Mecanismo de alocação de memória da JVM; Introdução à ferramenta de desenvolvimento Eclipse; Depuração; Análise de Algoritmos e Complexidade Assintótica; *HashMap*; *Streams*; Serialização.

5 Objetivos

Geral: Auxiliar na aprendizagem das disciplinas de Programação Orientada a Objetos 1 e 2, produzindo um projeto em console Java, seguido de sua adaptação para interface gráfica, mostrando normas e passos para um código bem feito e reutilizável.

Específico: São objetivos do minicurso:

- Apresentar as especificidades básicas da linguagem Java, dentre elas:
  - Classe principal e método `main`;
  - Formas de declarar/instanciar variáveis e vetores;
  - Ciclo de vida das variáveis;
  - Referência e cópia;
  - Funcionamento do método construtor.
- Criar um projeto simples para mostrar o uso das estruturas básicas (comparação, seleção, repetição, funções...) da programação imperativa;
- Melhorar o desempenho e entendimento dos alunos nas disciplinas de Programação Orientada a Objetos 1 e 2;
- Instruir os participantes a respeito das boas práticas de programação, a fim de que façam códigos bem estruturados, reutilizáveis e legíveis, aplicando conceitos de código-limpo e que façam escolhas adequadas de padrões de projetos;
- Introduzir noções de algoritmos e resolução de problemas computacionalmente;
- Criar um projeto com enfoque no interfaceamento gráfico e uso de diferentes recursos de programação orientada a objetos (interfaces, classes, classes abstratas...);
- Expor formas de melhorar a manutenibilidade de software.

6 Conteúdo Programático

1. Introdução a Java (2 horas):
   (a) Características principais da linguagem;
   (b) Processo de compilação e *Bytecode*;
   (c) Processo de execução da JVM (*Java Virtual Machine*);
   (d) Introdução ao paradigma Imperativo;
   (e) Tipos primitivos;
   (f) Estruturas condicionais: `if`.
2. Coleções (2 horas):

(a) Laços de repetição;
(b) *Raw-arrays*;
(c) Tempo de vida de variáveis: Escopo;
(d) Matrizes;
(e) *Cache-friendliness*.

3. Tipos definidos por usuário (2 horas):
   (a) Classes e objetos;
   (b) Atributos;
   (c) Alocação de memória na JVM;
   (d) Garbage-Collection;
   (e) Tempo de vida de variáveis: Objetos;
   (f) Elementos estáticos.
4. Introdução a Algoritmos (2 horas):
   (a) Introdução a funções e procedimentos;
   (b) Busca linear e *Binary-Search*;
   (c) Complexidade de Algoritmos;
   (d) Notação assintótica;
   (e) *Binary-Sort*.
5. Métodos, funções e procedimentos (2 horas):
   (a) Cópia e Referência;
   (b) Depuração de código;
   (c) Encapsulamento de atributos;
   (d) Boas práticas: API.
6. Padrões de Projeto (2 horas):
   (a) Diagrama de Casos de Uso;
   (b) Padrão MVC;
   (c) *HashMap*;
   (d) Diagrama de Classes.
7. Persistência (2 horas):
   (a) *Streams*;
   (b) Serialização;
   (c) Dependência entre objetos.

8. Interface Gráfica (2 horas):
    (a) Componentes básicos AWT/`javax.swing`;
    (b) *GridLayout*;
    (c) Padrão Observador-Observável;
    (d) *Callback*;
    (e) Funções *lambda*;
    (f) Classes anônimas.
9. Gráficos e renderização (2 horas):
    (a) Enumeradores;
    (b) Estrutura de seleção;
    (c) Cores RGB;
    (d) *Bitmaps*;
    (e) *Double-Buffering*;
    (f) *Page-Flipping*.
10. Introdução a Estruturas de Dados (2 horas):
    (a) *Generics*;
    (b) Lista com vetor;
    (c) Lista encadeada;
    (d) Interfaces;
    (e) Classes abstratas;
    (f) Pilha;

7 Cronograma

(a) Primeiro encontro:
    (a) Introdução a Java;
    (b) Apresentação do material de apoio.
(b) Segundo encontro:
    (a) Coleções.
(c) Terceiro encontro:
    (a) Tipos definidos por usuário.
(d) Quarto encontro:

    (a) Introdução a Algoritmos.

(e) Quinto encontro:

    (a) Métodos, funções e procedimentos.

(f) Sexto encontro:

    (a) Padrões de Projeto;
    (b) Menção a padrões de projeto não abordados na aula.

(g) Sétimo encontro:

    (a) Persistência.

(h) Oitavo encontro:

    (a) Interface Gráfica.

(i) Nono encontro:

    (a) Gráficos e renderização.

(j) Décimo encontro:

    (a) Introdução a Estruturas de Dados;
    (b) Conclusões e menções a respeito do criticismo de Java.

Referências

[1] Oracle. Java Platform SE 8 Documentation. https://docs.oracle.com/javase/8/docs/api/index.html.

[2] Thomas H. Cormen, Clifford Stein, Ronald L. Rivest, and Charles E. Leiserson. *Introduction to Algorithms*. McGraw-Hill Higher Education, 2nd edition, 2001.

[3] Raul Sidnei Wazlawick. *Engenharia de Software para Sistemas de Informação: Conceitos e práticas que fazem sentido*. 2012.