FACAMP
FACULDADES DE CAMPINAS

Laboratório de Programação

Aula 5

Programação GUI (Java/Swing), eventos e tratamento de exceções

2º semestre de 2019

Prof José Martins Junior

AWT

- O AWT (Abstract Window Toolkit) é uma biblioteca de classes para programação GUI básica
  - Foi lançada pela Sun com o Java 1.0
  - Opera delegando a criação de elementos de interface e seus comportamentos ao Kit GUI nativo, em cada plataforma
- Metodologia baseada em semelhantes (*peers*) nativos
  - Dificuldades com a portabilidade
  - Aparência diferente de outros aplicativos, nas diferentes plataformas (Windows, Unix, Mac..)
  - O mesmo programa apresentava erros diferentes quando compilado em cada uma das plataformas

Swing

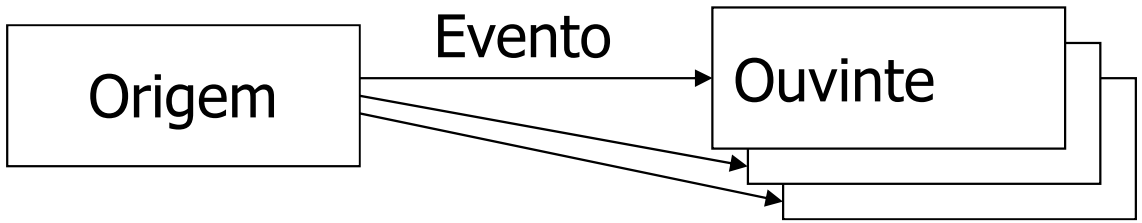
- Em 1996 a Netscape lançou uma biblioteca GUI, a IFC (Internet Foundation Classes)
  - Elementos desenhados em janelas em branco
  - A única funcionalidade *peer* necessária era a maneira de formar janelas e desenhar nelas
  - Os componentes da Netscape se pareciam e se comportavam da mesma forma em diferentes plataformas
- A Sun, junto com a Netscape, criou a JFC (Java Foundation Classes), contendo
  - Swing - Toolkit GUI não baseado em semelhantes
  - APIs de acessibilidade, 2D e de “arrastar e soltar”
- O conjunto Swing foi integrado ao núcleo do Java 2
  - Pacote chama-se javax.swing, mantendo compatibilidade com o Java 1.1 (nele, o Swing pode ser integrado como extensão)

Frames e Janelas

- Frames são janelas de nível mais alto
  - São containers que podem conter outros objetos da interface
- Em Swing - JFrame
  - Estende a classe Frame (AWT) e, portanto, baseia-se em *peers*
  - Em Swing: letra J precede o nome da classe equivalente em AWT
- Passos mínimos para a criação e exibição de um frame
  - Estender a classe JFrame
    - Ex.: class FirstFrame extends JFrame
  - Dimensionar o objeto (do contrário, terá dimensões 0x0)
    - Ex.: (No construtor) setSize(300,200); //de Component
  - Criação do objeto
    - Ex.: JFrame frame = new FirstFrame();
  - Tornar visível e trazer para a frente
    - Ex.: frame.setVisible(true); //de Window

Eventos no Java 1.1

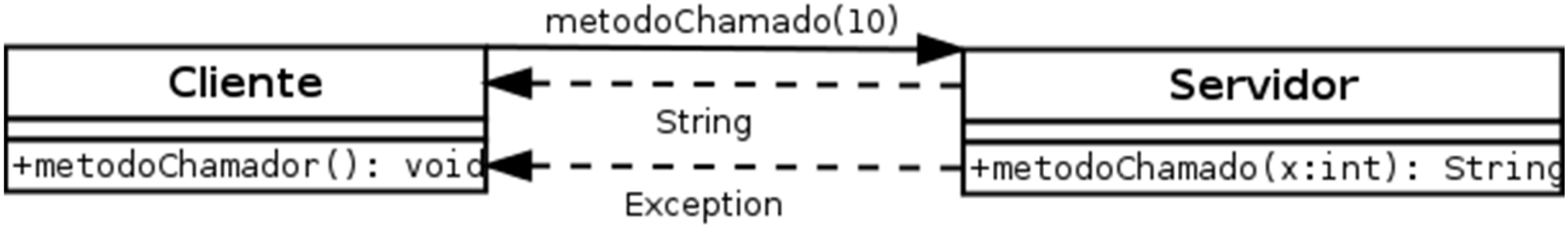
- A versão 1.1 de Java trouxe mudanças significativas no tratamento de eventos pela AWT (Swing usa tal versão)
- Modelo de eventos em Java



- Objetos envolvidos
  - Origem - registra, através de métodos próprios, objetos ouvintes para determinado evento. Ex.: origem.addActionListener(ouvinte)
  - Eventos - objetos enviados aos ouvintes registrados, quando o evento ocorre
  - Ouvinte - instância de objeto que implementa uma listener interface compatível com o tipo de evento envolvido. Deve implementar cada método da interface ouvinte que será chamado pela origem para a passagem dos eventos, podendo assim tratá-los

Exceções

- Java possui recursos para manipulação de erros ou exceções
  - Estabelece um fluxo alternativo para retorno de um método, em regime de exceção



- Alguns exemplos de erros
  - Erros de parâmetros de entrada (formato incorreto de números, p.e.)
  - Erros em acesso a dispositivos (I/O de arquivos e rede, p.e.)
    - Limitações físicas (memória, tamanhos de Strings e arrays, p.e.)
  - Erros de código (casting inapropriado, objetos nulos, p.e.)

Tipos de exceções

- Toda exceção em Java é uma subclasse de `Throwable`
- Existem dois filhos diretos dessa classe
  - `Error`
    - Erros internos da JVM, como a falta de recursos no sistema, em tempo de execução
    - Não podem ser previstos, tratados ou lançados
    - Exemplos: `AWTError`, `VirtualMachineError`
  - `Exception`
    - Define uma hierarquia de exceções que podem/devem ser previstas, lançadas ou tratadas
    - Existem dois tipos: runtime (unchecked) e checked exceptions

Exceptions

- `RuntimeException`
  - Exceções em tempo de execução e, portanto, não são verificadas pelo compilador (conhecidas como **unchecked** exceptions)
  - Como consequência, não precisam ser obrigatoriamente anunciadas e podem ser lançadas ou obtidas/tratadas independentemente disso
  - Exemplos: `NullPointerException`, `ClassCastException`, `IndexOutOfBoundsException`
- **Checked** exceptions
  - Representam exceções previsíveis, de acordo com a situação (acesso a arquivos, conexão com BD ou rede, p.e.)
  - São verificadas pelo compilador e, portanto, **devem ser anunciadas** e obtidas/tratadas ou repassadas
  - Exemplos: `FileNotFoundException`, `SQLException`, `NumberFormatException`

Exceções

- Exceções em métodos
  - Podem (no caso das checked exceptions, devem) ser **anunciadas** (`throws`) e definem um caminho alternativo à execução do método chamado
  - Quando o método em execução se deparar com a ocorrência da situação prevista, ele pode **lançar** (`throw`) a exceção, o que interrompe sua execução naquele momento
  - O chamador pode **tentar** (`try`) executar o método que anuncia a exceção, em caso dela ocorrer, pode **pegá-la** (`catch`) e tratá-la
  - Alternativamente, o método chamador pode também anunciá-la (`throws`), e **repassar** a obrigação para o método que o chamar

Bibliografia

DEITEL, P.; DEITEL, H. Java TM: como programar. 8ª edição. São Paulo: Pearson Prentice Hall, 2012. 1144p.

GOSLING, J.; ARNOLD, K.; HOLMES, D. A Linguagem de Programação Java. 4ª edição, Porto Alegre: Bookman, 2007.