WEBIDE 2.0

Um ambiente web usando Gwt-Ext para acompanhamento e desenvolvimento de exercícios de programação

Rafael Adriano

Orientador
Prof. Adilson Vahldick

Roteiro

- Introdução
- Fundamentação teórica
- Desenvolvimento
- Conclusão
- Extensões

Introdução

- Objetivos do trabalho
  - Disponibilizar uma nova interface mais amigável usando o framework Gwt-Ext;
  - Utilizar tecnologia de depuração com a arquitetura JPDA;
  - Fornecer recursos de edição como número de linhas, content assist/auto complemento e syntax highlighting.

Fundamentação Teórica

- GWT
- EXT JS
- Gwt-Ext
- JPDA
- Trabalhos correlatos

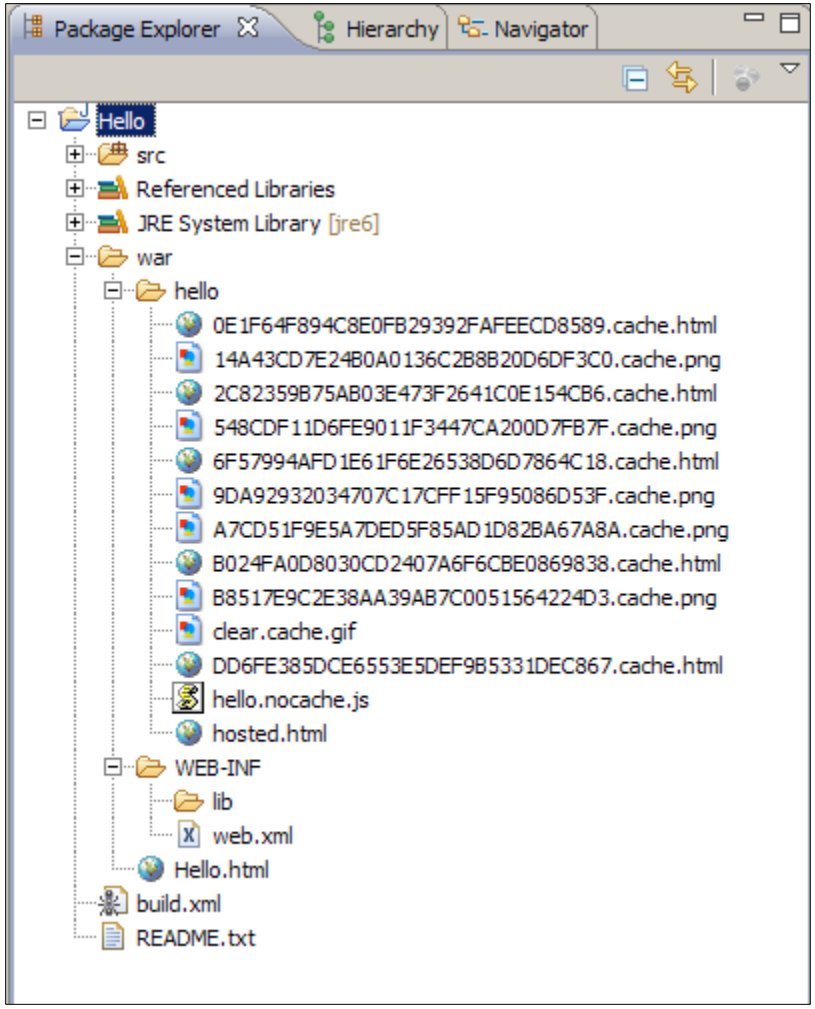
Fundamentação Teórica

- GWT
  - Fácil desenvolvimento de aplicações web
    - Reutilização de código
    - Fácil manutenção

Fundamentação Teórica

- GWT
  - Características
    - Compilador
      - Java -> JavaScript
    - Comunicação
      - XML
      - JSON
      - GWT RPC
    - JSNI

Fundamentação Teórica



Fundamentação Teórica

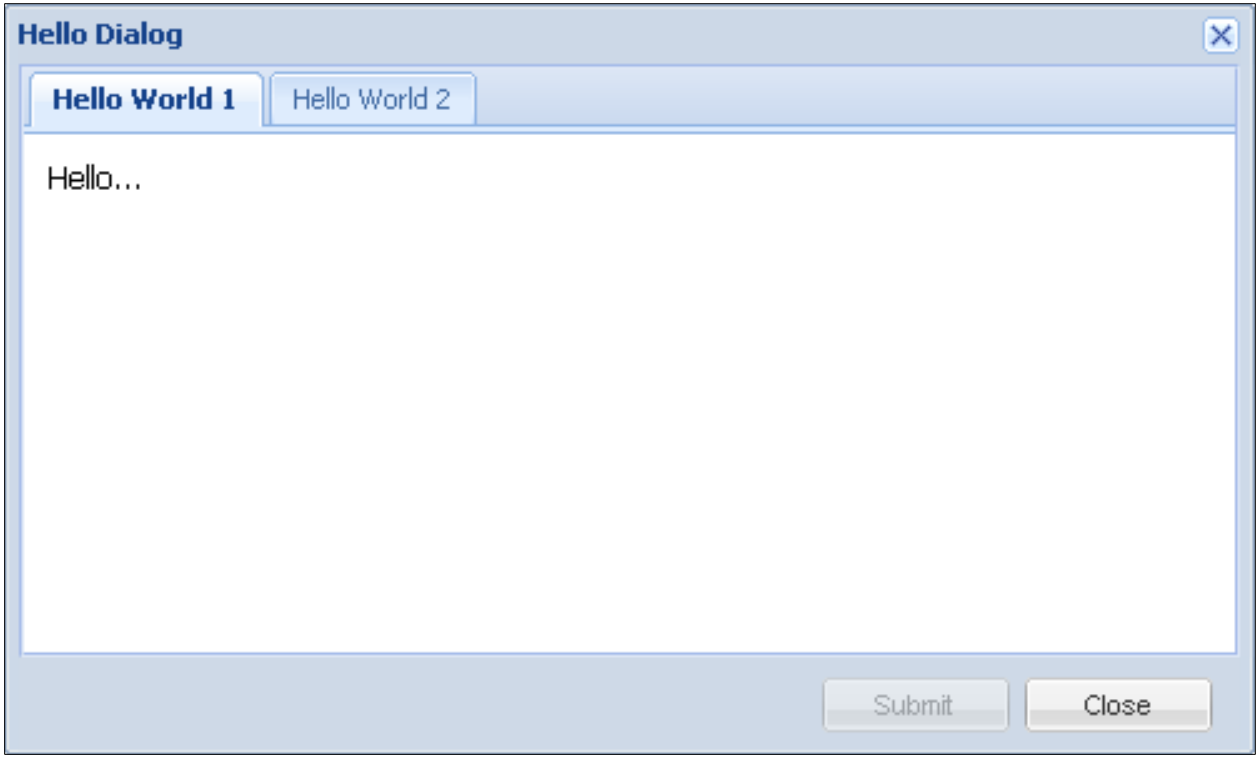
```
private native JavaScriptObject getCaretPosition() /*-{
     var field =
     this.@com.gwtext.client.widgets.Component::getOrCreateJsObj()();
     winFrame = field.iframe.contentWindow;
     var result = {offset: 0};

     if (select.ie_selection) { // IE
          var cursor = select.cursorPos(winFrame.document.body, true);
          if (cursor) {
                    result.offset = cursor.offset;
          }
     } else {
          var cursor = select.cursorPos(winFrame.document.body, true);
          if (cursor) {
                    result.offset = cursor.offset;
          }
     }
     return result;
}-*/;
```

Fundamentação Teórica

- EXT JS
  - Interfaces ricas
  - Alto desempenho
  - Componentes personalizáveis
  - Widgets

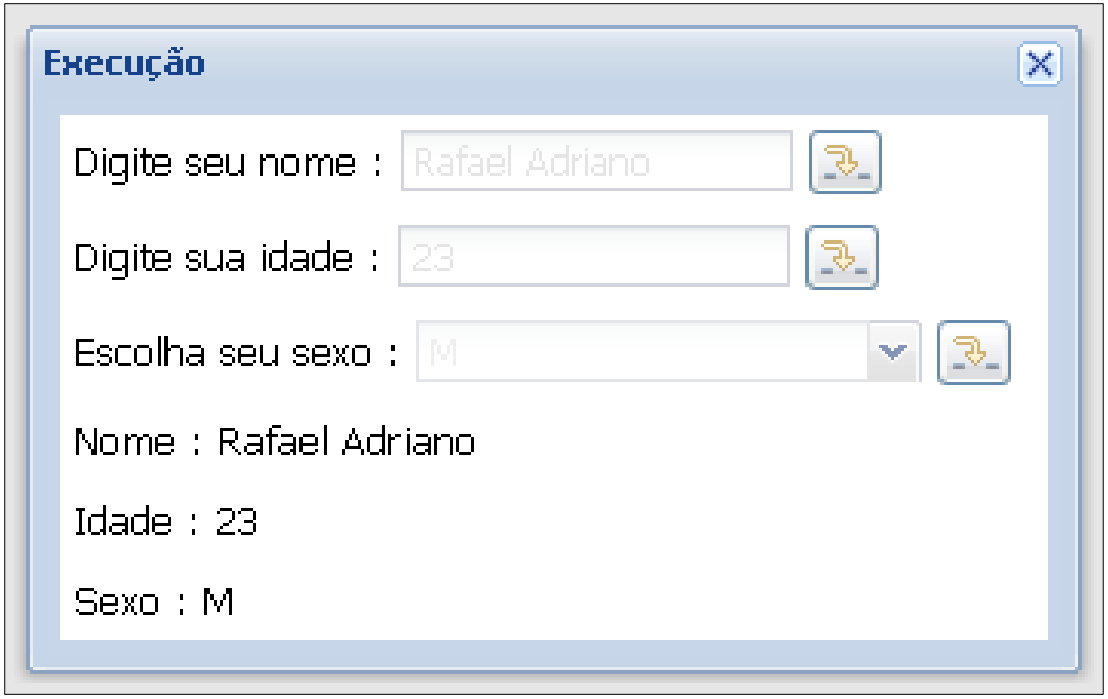
Fundamentação Teórica



Fundamentação Teórica

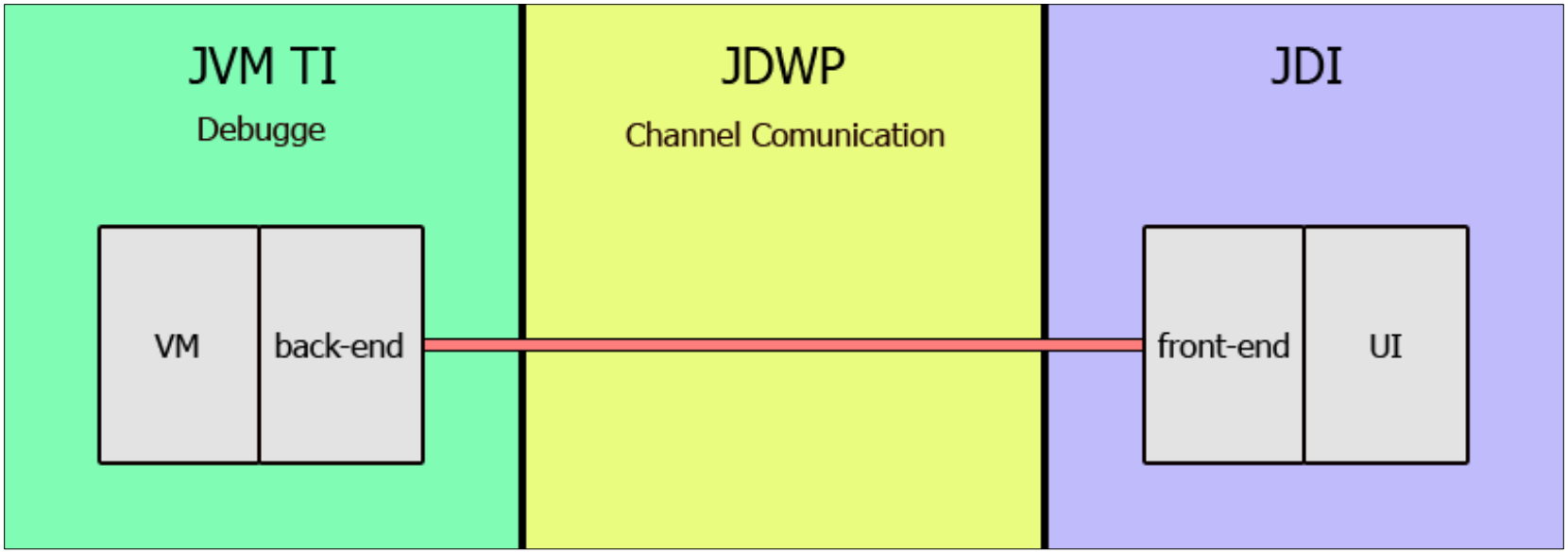
- Gwt-Ext
  - Open Source
  - União GWT + Ext JS
  - Utiliza JSNI

Fundamentação Teórica



Fundamentação Teórica

- JPDA



Fundamentação Teórica

- JPDA
  - JVMTI
    - Nativa
    - Recebe requisições
    - Notifica eventos

Fundamentação Teórica

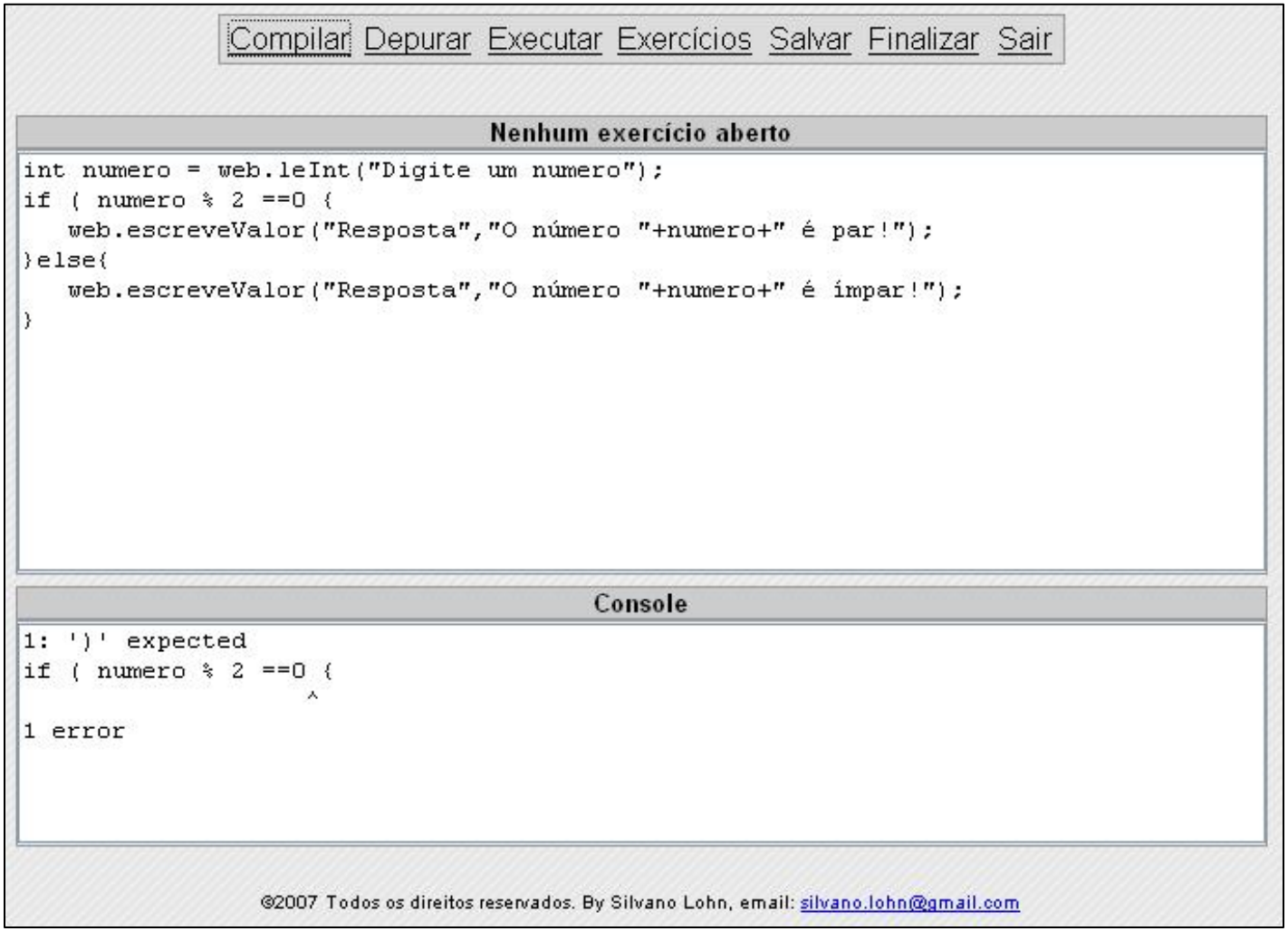
- JPDA
  - JDWP
    - Formato das informações
    - Transporte das Requisições/Eventos
    - Não especifica protocolo

Fundamentação Teórica

- JPDA
  - JDI
    - Alto nível
    - Recebe eventos
    - Envia requisições

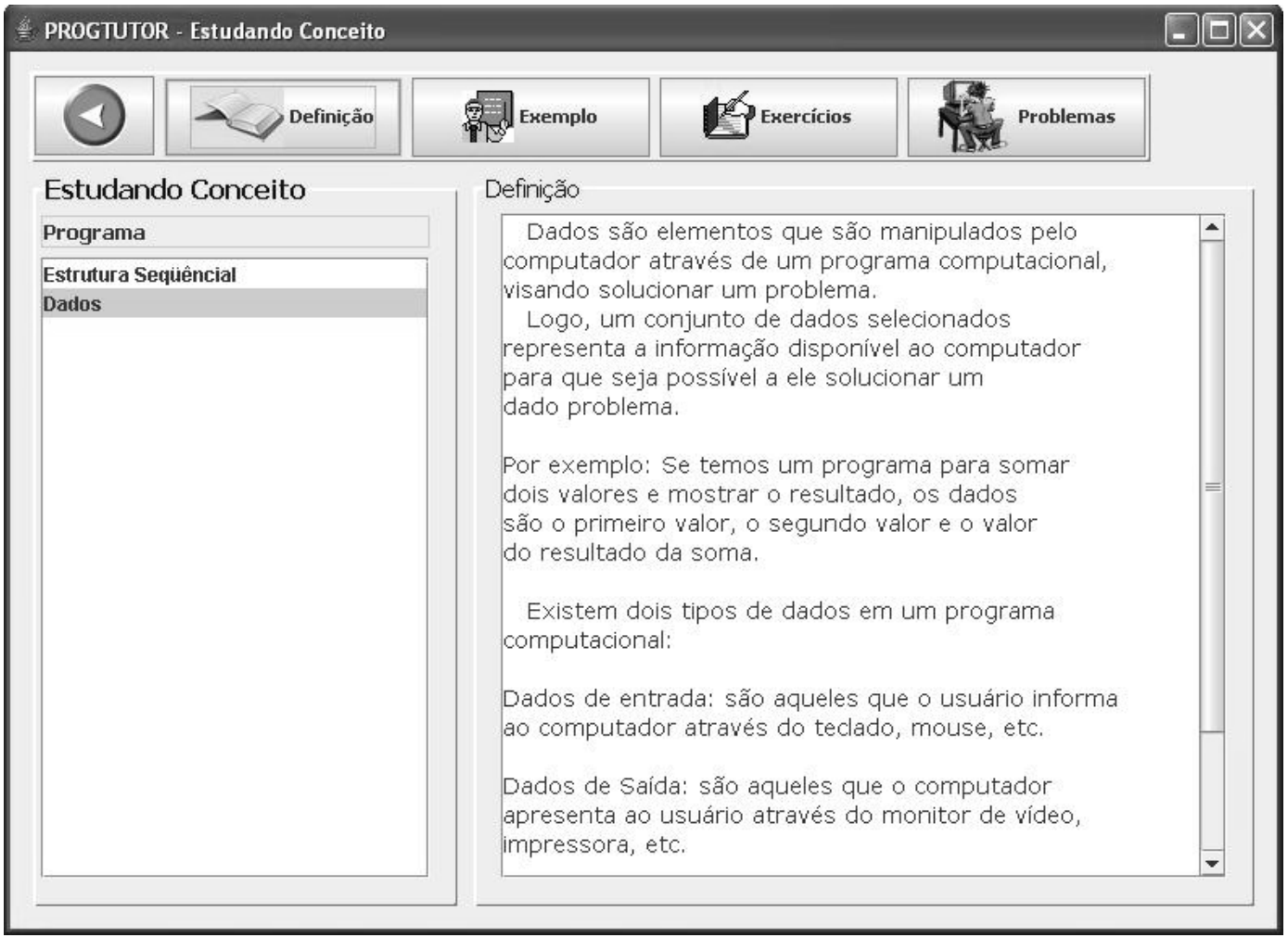
Fundamentação Teórica

- Trabalhos correlatos
  - Ambiente na Web para Execução e Depuração de Programas com Sintaxe Java (LOHN, 2008)
    - WebIde 1.0
    - Edição de código
    - Depuração e execução de código
    - Ambiente administrativo para exercícios


Compilar Depurar Executar Exercícios Salvar Finalizar Sair
Nenhum exercício aberto
int numero = web.leInt("Digite um numero");
if (numero % 2 ==0 {
web.escreveValor("Resposta","O número "+numero+" é par!");
}else{
web.escreveValor("Resposta","O número "+numero+" é ímpar!");
}
Console
1: ')' expected
if (numero % 2 ==0 {
^
1 error
©2007 Todos os direitos reservados. By Silvano Lohn, email: silvano.lohn@gmail.com

Fundamentação Teórica

- Trabalhos correlatos
  - Ferramenta Computacional de Apoio ao Processo de Ensino-Aprendizagem dos Fundamentos de Programação de Computadores (BRANCO; SCHUVARTZ, 2007)
    - Apoio a aprendizagem
    - Utiliza Inteligência Artificial
    - Módulo domínio, aluno e tutor


PROGTUTOR - Estudando Conceito
Definição
Exemplo
Exercícios
Problemas
Estudando Conceito
Programa
Estrutura Seqüêncial
Dados
Definição
Dados são elementos que são manipulados pelo computador através de um programa computacional, visando solucionar um problema.
Logo, um conjunto de dados selecionados representa a informação disponível ao computador para que seja possível a ele solucionar um dado problema.
Por exemplo: Se temos um programa para somar dois valores e mostrar o resultado, os dados são o primeiro valor, o segundo valor e o valor do resultado da soma.
Existem dois tipos de dados em um programa computacional:
Dados de entrada: são aqueles que o usuário informa ao computador através do teclado, mouse, etc.
Dados de Saída: são aqueles que o computador apresenta ao usuário através do monitor de vídeo, impressora, etc.

Fundamentação Teórica

- Trabalhos correlatos
  - Um Ambiente para Ensino de Programação com Feedback Automático de Exercícios (HINTERHOLZ JR, 2009)
    - Estudar técnicas de avaliação de algoritmos
    - Regressão linear múltipla
    - N-grama

Desenvolvimento

- Requisitos
- Especificação
- Implementação
- Resultados e Discussão

Desenvolvimento

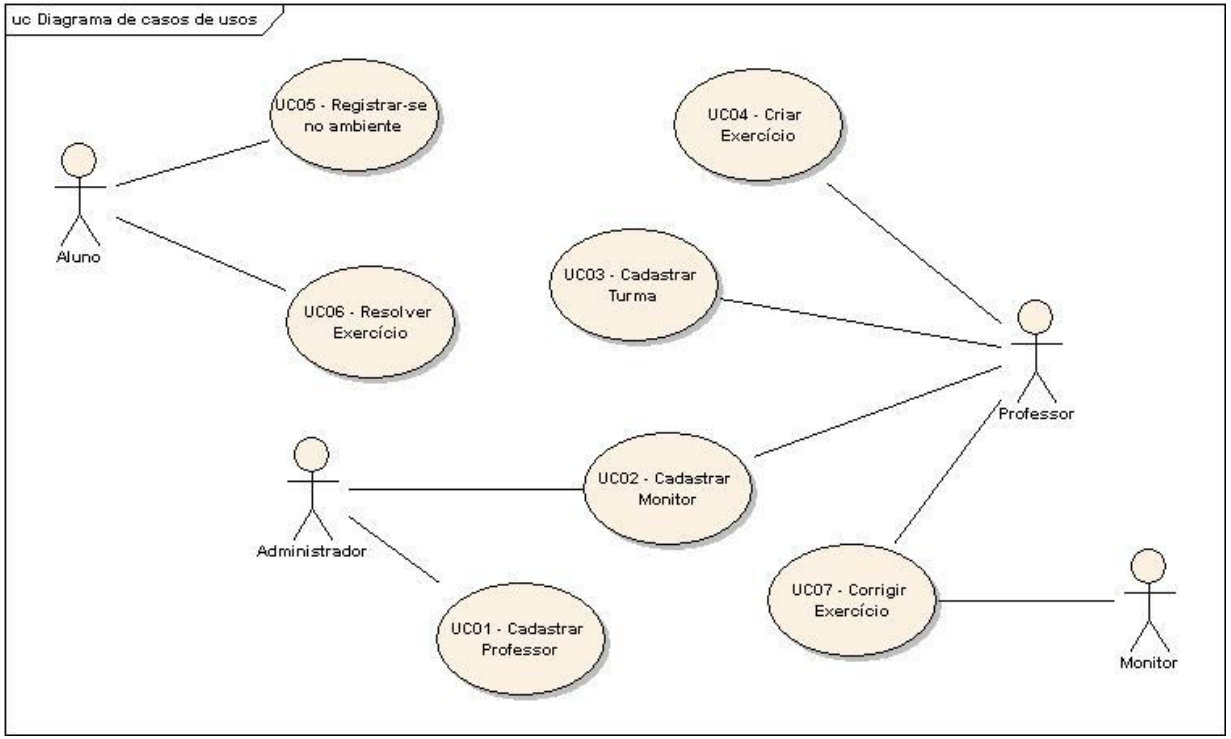
- Requisitos
  - permitir que o programa seja carregado, editado e salvo (RF);
  - fazer a execução do programa (RF);
  - fazer a depuração do programa de modo passo-a-passo informando a linha e o valor das variáveis em execução (RF);
  - fazer a compilação do programa informando quando houver os erros de compilação (RF);
  - fazer um ambiente administrativo onde seja possível criar e corrigir os exercícios (RF);
  - permitir que os usuários cadastrem-se no ambiente (RF);

Desenvolvimento

- Requisitos
  - permitir o content assist/auto-complemento de código (RF);
  - fazer o syntax highlighting no código (RF);
  - apresentar numeração de linhas (RF);
  - utilizar a arquitetura JPDA para a depuração (RNF);
  - ser implementado utilizando o framework GWT-Ext (RNF);
  - ser compatível com os principais navegadores comerciais (RNF).

Desenvolvimento

- Especificação

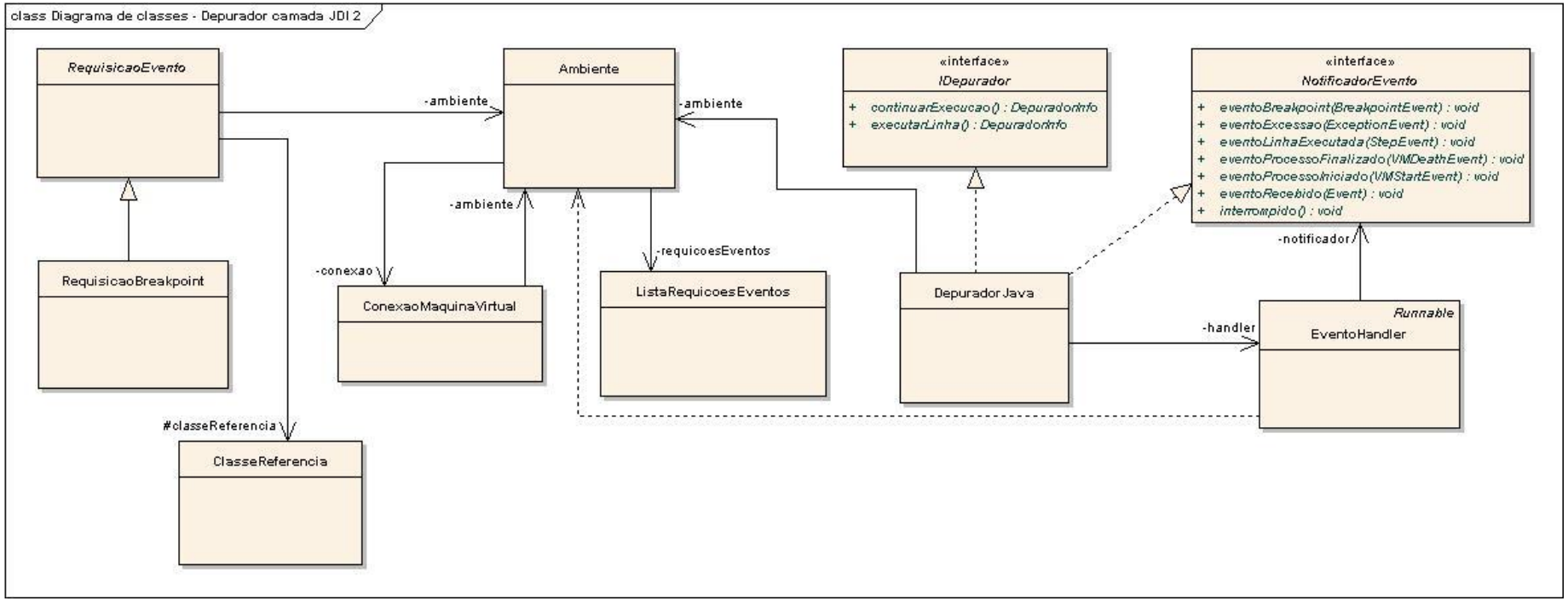


Desenvolvimento

- Especificação

Desenvolvimento

- Especificação



Desenvolvimento

- Implementação – JPDA
  - Execução de linha

```
boolean executou = GerenciadorComando.comandoExecutarLinha(codigoUsuario);
if (executou) {
        esperar();
}
```

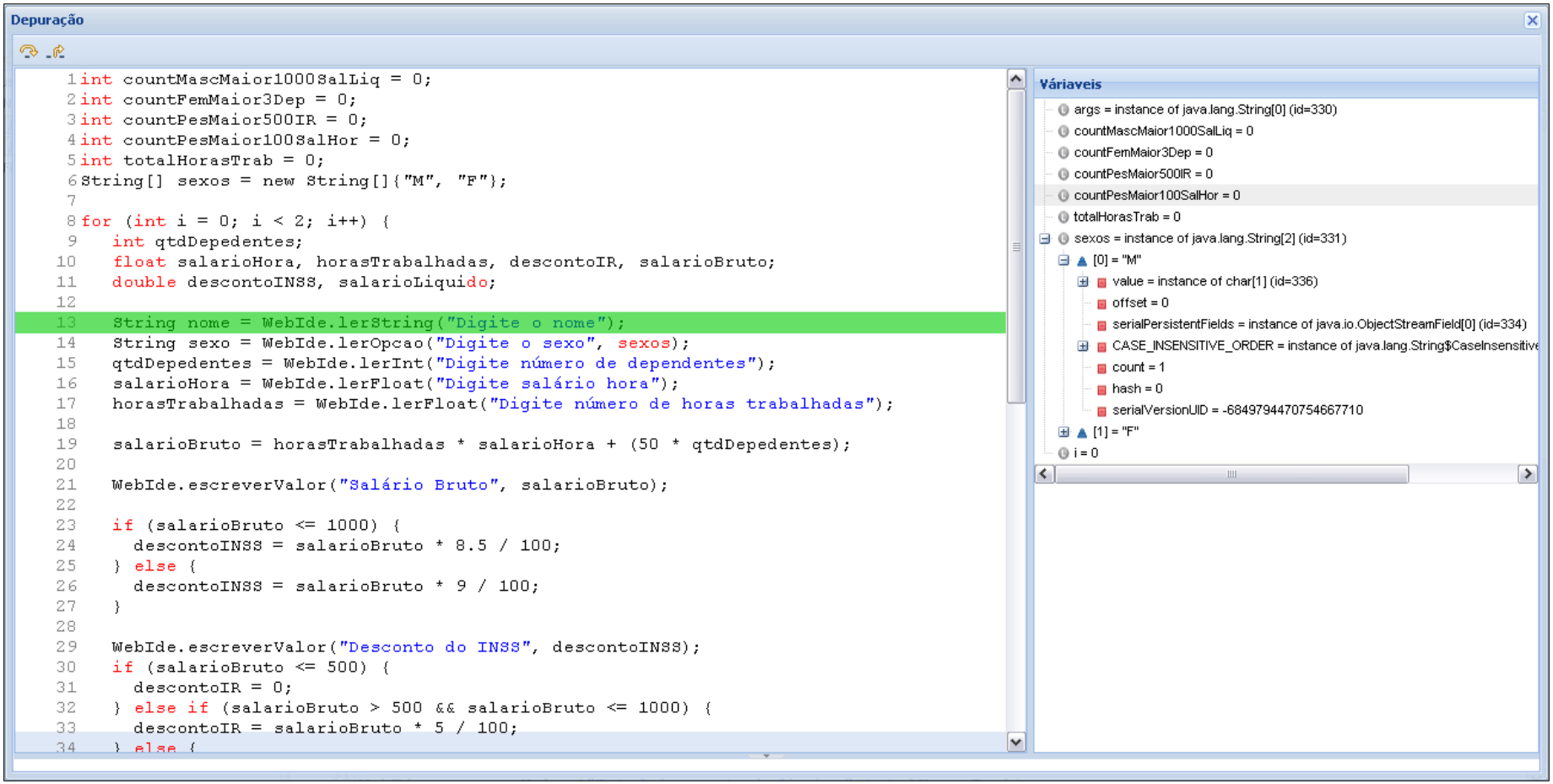
```
Ambiente ambiente = AmbienteManager.getAmbiente(codigoUsuario);
EventRequestManager reqMgr =
ambiente.getMaquinaVirtual().eventRequestManager();
StepRequest request = reqMgr.createStepRequest(threadUsuario,
StepRequest.STEP_LINE, StepRequest.STEP_OVER);
request.addCountFilter(1);
request.enable();
GerenciadorThreadsUsuarios.invalidarThreadsUsuario(codigoUsuario);
ambiente.getMaquinaVirtual().resume();
```

Desenvolvimento

- Implementação – JPDA
  - Carga de váriaveis

```
...
List<LocalVariable> variables = frame.visibleVariables();
List<Variavel> vars = new ArrayList<Variavel>();
for (LocalVariable localVariable : variables) {
        Variavel var = new Variavel();
        Value value = frame.getValue(localVariable);
        var.setNome(localVariable.name());
        if (value != null) {
                var.setValor(value.toString());
                // Verifica se não tem subvariaveis
                loadValue(value, var);
        }
        vars.add(var);
}
...
```

Desenvolvimento



Desenvolvimento

- Implementação – Syntax Highlighting
  - Expressões regulares

```
"([^"\\\n\r]|(\\([ntbrf\\'"]|[0-7]([0-7])?|[0-3][0-7][0-7]))|(\\u[0-9a-fA-F][0-9A-Fa-f][0-9A-Fa-f][0-9A-Fa-f]))*"
```

```
//([^\n\r])*(\n|\r|\r\n)?
```

Desenvolvimento

- Implementação – Syntax Highlighting
  - Compilação das expressões

```
// Expressões regulares
this.comentarioBloco = Pattern.compile(comentarioBloco);
this.comentarioLinha = Pattern.compile(comentarioLinha);
this.valoresLiterais = new Pattern[valoresLiterais.length];
// Valores literais
for (int i = 0; i < valoresLiterais.length; i++) {
        this.valoresLiterais[i] = Pattern.compile(valoresLiterais[i]);
}
// Palavras reservadas
StringBuilder bfPalavrasReservadas = new StringBuilder();
for (int i = 0; i < palavrasReservadas.length; i++) {
        bfPalavrasReservadas.append(palavrasReservadas[i]);
        bfPalavrasReservadas.append("|");
}
bfPalavrasReservadas.setLength(bfPalavrasReservadas.length() - 1);
this.palavrasReservadas = Pattern.compile(bfPalavrasReservadas.toString());
```

Desenvolvimento

WebIde 2.0

SALÁRIO LÍQUIDO DE FUNCIONÁRIOS

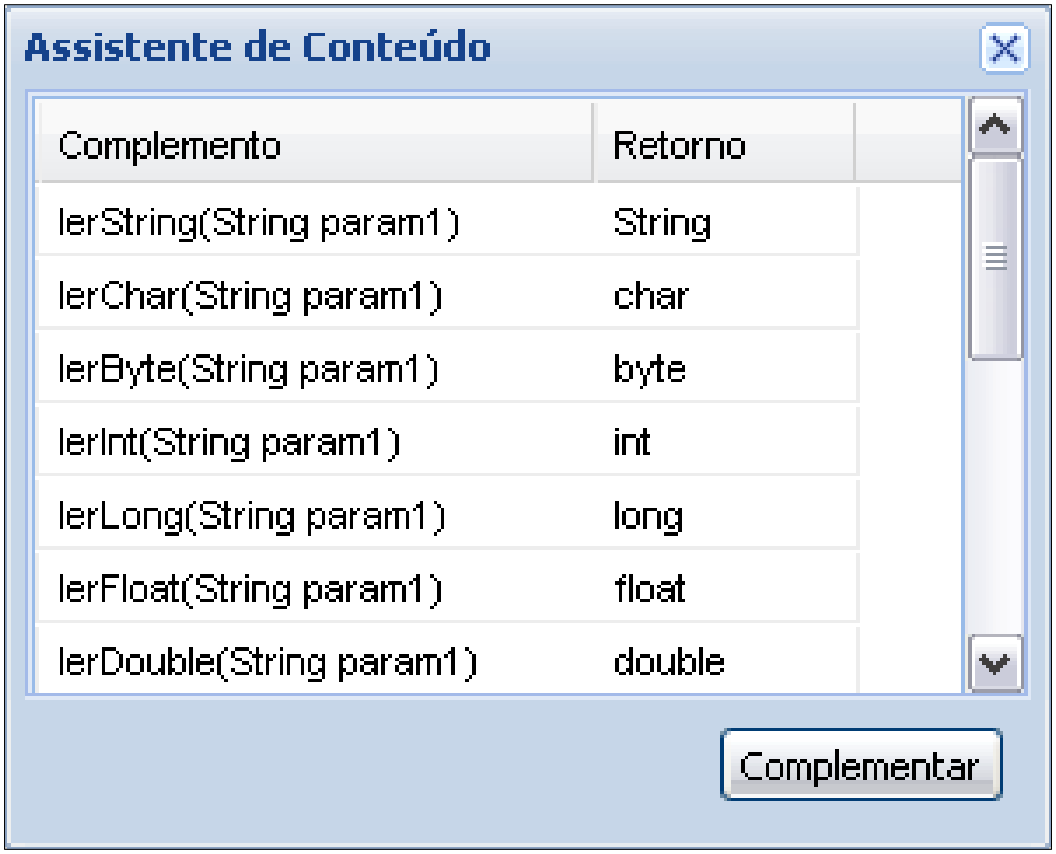
```
1  int countMascMaior1000SalLiq = 0;
2  int countFemMaior3Dep = 0;
3  int countPesMaior500IR = 0;
4  int countPesMaior100SalHor = 0;
5  int totalHorasTrab = 0;
6  String[] sexos = new String[]{"M", "F"};
7
8  for (int i = 0; i < 2; i++) {
9    int qtdDepedentes;
10   float salarioHora, horasTrabalhadas, descontoIR, salarioBruto;
11   double descontoINSS, salarioLiq;
12
13   String nome = WebIde.lerString("Digite o nome");
14   String sexo = WebIde.lerOpcao("Digite o sexo", sexos);
15   qtdDepedentes = WebIde.lerInt("Digite número de dependentes");
16   salarioHora = WebIde.lerFloat("Digite salário hora");
17   horasTrabalhadas = WebIde.lerFloat("Digite número de horas trabalhadas");
18
19   salarioBruto = horasTrabalhadas * salarioHora + (50 * qtdDepedentes);
20
21   WebIde.escreverValor("Salário Bruto", salarioBruto);
22
23   if (salarioBruto <= 1000) {
24      descontoINSS = salarioBruto * 8.5 / 100;
25   } else {
26      descontoINSS = salarioBruto * 9 / 100;
27   }
28
29   WebIde.escreverValor("Desconto do INSS", descontoINSS);
30   if (salarioBruto <= 500) {
31      descontoIR = 0;
32   } else if (salarioBruto > 500 && salarioBruto <= 1000) {
33      descontoIR = salarioBruto * 5 / 100;
```

Desenvolvimento

- Implementação – Content Assist

```
List<Complementar> complementares = Collections.emptyList();
if (coluna < input.length()) {
        StringBuilder sb = new StringBuilder(input);
        sb.replace(coluna, sb.length()-1, DUMMY_TEMP + "();");
        CompilationUnit compUnit = innerParser(sb.toString());
        if (compUnit != null) {
                VisitadorJavaBind visitador = new VisitadorJavaBind();
                compUnit.accept(visitador, null);
                complementares = visitador.getComplementares();
        }
} else if (coluna == input.length()) {
        CompilationUnit compUnit = innerParser(input + DUMMY_TEMP +
"();");
        if (compUnit != null) {
                VisitadorJavaBind visitador = new VisitadorJavaBind();
                compUnit.accept(visitador, null);
                complementares = visitador.getComplementares();
        }
}
```

Desenvolvimento



Desenvolvimento

- Estudo de Caso
  - Implantação
    - Criação da base de dados
    - Instalação da aplicação web no TomCat
    - Configurações das propriedades
      - Base de dados
      - Diretórios
      - Caminho da instalação dos executáveis java e javac

Desenvolvimento

- Estudo de Caso
  - Validação
    - Testes de compatibilidade entre os navegadores
    - Testes de compilação, execução e depuração
    - Testes de usabilidade do sistema
    - Testes multiusuário
  - Problemas ocorridos
    - O processo da aplicação travou

Desenvolvimento

- Resultados e Discussão
  - Poucas dúvidas na parte dos alunos
  - Problemas no editor durante validação
  - Dificuldade na interpretação da mensagem de erro
    - Adicionado na ajuda erros mais comuns
  - Notificação quando o exercício era reaberto

Desenvolvimento

- Resultados e Discussão
  - Resultado da captura de ações dos usuários

Descrição	Resultado
1. Quantidade de alunos que usaram o ambiente	23 alunos
2. Média de tempo gasto pelos alunos para resolver e finalizar o exercício	81 horas
3. O maior tempo gasto por um aluno para resolver e finalizar o exercício	195 horas
4. O menor tempo gasto por um aluno para resolver e finalizar o exercício	1 hora
5. Quantidade de chamadas para abrir a ajuda de contexto	289
6. Quantidade de chamadas para execução	1465 execuções
7. Quantidade de chamadas para depuração	190 depurações
8. Quantidade de reabertura de exercícios	12 reaberturas
9. Quantidade de vezes que os exercícios foram salvos	673
10. Quantidade de vezes que abriram o enunciado do exercício	394

Desenvolvimento

- Resultados e Discussão
  - Comparação com trabalhos correlatos

Descrição	WebIde 1.0	WebIde 2.0
Compilação de programas	X	X
Execução de programas	X	X
Depuração de programas	X	X
Alteração de variáveis em tempo de depuração	X	
Programação em Java	X	X
Ambiente integrado para criação, resolução e correção de exercícios	X	X
Suporte a syntax highlighting		X
Suporte a content assist		X
Numeração de linhas		X
Interface rica de fácil usabilidade		X

Desenvolvimento

- Resultados e Discussão
  - Comparação com trabalhos correlatos

Descrição	Ferramenta Computacional de Apoio ao Processo de Ensino-Aprendizagem dos Fundamentos de Programação de Computadores	WebIde 2.0
Indicar o melhor caminho para aluno	Inteligência Artificial	Intervenção do Professor
Aplicação totalmente web		X
Exemplos ilustrativos	X	
Editor com recursos de edição		X

Desenvolvimento

- Resultados e Discussão
  - Comparação com trabalhos correlatos

Descrição	Um Ambiente para Ensino de Programação com Feedback Automático de Exercícios	WebIde 2.0
Correção automática de exercícios	X	
Ambiente administrativo de exercícios	X	X
Compilação de algoritmo		X
Depuração de algoritmo		X

Conclusão

- Os objetivos foram atingidos, somente o recurso content assist não funciona no navegador Internet Explorer
- Ambiente estável
- Fácil implantação
- Interface amigável
- Ótima aceitação dos alunos e do professor

Extensões

- Técnica Sand Box, para capturar processos que podem travar o servidor
- Enunciado inteligente, onde o professor possa criar regras de programação no exercício, e a aplicação guie o aluno a programar no caminho correto
- Assistente, que trate erros de compilação, e explique para o aluno porque esse erro pode ocorrer e como ele pode ser corrigido
- Mais funcionalidades na depuração, como colocar breakpoints, mudar valor da variável, executar expressões ou fazer drop to frame
- Maneiras para o professor e o monitor reutilizarem os seus comentários para problemas comuns encontrados nos exercícios dos alunos
- Mecanismo de correção automática dos exercícios usando as ideias do trabalho de Moreira e Favero (2009)

Apresentação da Aplicação