

# Smalg Platform: Uma Plataforma Educacional

Adriner Maranhão de Andrade  
Dalton Solano dos Reis (Orientador)

# Roteiro

- Introdução
- Objetivos
- Fundamentação teórica
- Trabalhos correlatos
- Descrição
- Resultados e discussões
- Conclusões

# Introdução

- A partir de uma perspectiva educacional, a ciência da computação pode se demonstrar desafiadora.
- A maioria dos abandonos ocorrem nos dois primeiros anos de estudo.
- Problemas na compreensão conceitual de programação pode levar os estudantes a sérios equívocos.

# Introdução

- A utilização de ferramentas visuais que ilustram os conceitos de programação e o funcionamento do código podem ser úteis no processo educacional.
- O uso da abstração, como interfaces, e conceitos encontrados em testes automatizados podem ser uma estratégia na formulação de uma ferramenta de ensino.

# Objetivos

- Desenvolver uma plataforma que servirá como uma ferramenta auxiliar no ensino de Ciência da Computação, em disciplinas como Algoritmo e Estrutura de dados.
  - Utilizar recursos de representação visual de execução.
  - Possibilitar a flexibilização do ensino ao professor.
  - Criar uma *engine* que gerará a partir do código desenvolvido uma representação visual de execução.
  - Definir formalmente uma estrutura que deverá ser utilizada para a codificação por parte do estudante.
  - Disponibilizar um material de apoio que servirá de documentação da plataforma.

# Fundamentação Teórica

- Dificuldades no ensino da computação
  - O aprendizado pode ser desafiante para os alunos, principalmente aos que estão nos anos iniciais e nunca tiveram contato com a área.
  - O maior índice de desistência se apresenta nos dois primeiros anos.
  - Alunos que apresentam um maior índice de esforço possuem maiores chances de permanecer nos estudos.

# Fundamentação Teórica

- Dificuldades no ensino da computação
  - Formulação de modelos mentais imprecisos podem dificultar o aprendizado.
  - Estudantes apresentam dificuldades em como desenvolver a solução para um problema e em como criar situações hipotéticas e excepcionais, para então encontrar situações de erro.
  - O primeiro passo para aprender programação normalmente é partindo de um pensamento humanizado, relacionado com a vida real, para então adentrar aos conceitos da programação.

# Fundamentação Teórica

- O uso da visualização no ensino
  - Muitos dos conteúdos centrais da área, estão relacionados com abstrações. Não é possível ver ou tocar um algoritmo ou uma estrutura de dados.
  - Uma visualização de um programa expõe o que normalmente está escondido e implícito.
  - Professores realizam ilustrações em sala para demonstrar conceitos. Automatizar esse processo, é otimizar a quantidade de cenários e o tempo investido.



# Fundamentação Teórica

- O uso da visualização no ensino
  - Alunos que se engajam com a ferramenta possuindo uma atividade mais ativa, apresentam melhores resultados do que os estudantes que utilizam a visualização somente passivamente.
  - O modo que o estudante interage com a visualização é mais importante do que as técnicas de visualização utilizadas.

# Fundamentação Teórica

- Abstração na programação e interfaces
  - No princípio a abstração tinha como propósito de gerar uma generalização descritiva dos dados de um programa de modo que pudessem ser trabalhados sem se preocupar em qual máquina seriam executados.
  - A abstração surge como uma forma de encapsulamento, ocultando informações específicas e disponibilizando informações genéricas.

# Fundamentação Teórica

- Abstração na programação e interfaces
  - Tipos de dados, como números e vetores, definem estruturas e operadores de como um dados será manipulado.
  - A inclusão da abstração em um tipo de dado caracteriza a possibilidade de uma de uma definição estrutural sem a necessidade de referenciar uma implementação real.
  - Dentro dos tipos de dados abstratos encontram-se as interfaces, que são especificações formais de comportamento. Sua estrutura é desacoplada de sua implementação.

# Fundamentação Teórica

- Conceitos relacionados a utilização de testes
  - A utilização de testes vai além de garantir somente a execução de um programa sem erros.
  - O conceito de *test-first* implicitamente define uma interface e uma especificação de comportamento a ser seguida, antes de uma implementação.

# Trabalhos correlatos

- Visualgo – Halim *et al.* (2012)
  - Tem como objetivo disponibilizar o aprendizado autodidático do aluno de vários algoritmos clássicos e não clássicos da área.
  - Possui acompanhamento passo a passo de execução.
  - Possui visualização gráfica da execução.
  - Apresenta uma plataforma unificada com diversos algoritmos diferentes.

# Trabalhos correlatos

- Visualgo – Halim *et al.* (2012)

The image displays a grid of 8 Visualgo algorithm cards, each with a unique color and diagram. The cards are arranged in two rows of four. Each card includes a title, a 'Training' tag, and a list of related tags.

- Sorting** (Green background): Diagram shows a bar chart with varying heights. Tags: array, algorithm, bubble, select.
- Bitmask** (Light Green background): Diagram shows a grid with 'AND' and binary digits. Tags: bit manipulation, set, cs3233, array.
- Linked List** (Blue background): Diagram shows a sequence of nodes labeled 'H' and 'T' connected by arrows. Tags: stack, queue, doubly, deque.
- Hash Table** (Orange background): Diagram shows a vertical bar and a table with indices 0-4. Tags: open addressing, linear, quadratic.
- Binary Heap** (Cyan background): Diagram shows a binary tree structure. Tags: priority queue, recursive, cs2010.
- Binary Search Tree** (Pink background): Diagram shows a binary search tree structure. Tags: adelson velskii landis, set, table, avl.
- Graph Structures** (Red background): Diagram shows a graph with nodes and edges. Tags: tree, complete, bipartite, dag.
- Union-Find DS** (Yellow background): Diagram shows a tree structure with nodes. Tags: path compression, disjoint, set.

# Trabalhos correlatos

- Visualgo – Halim *et al.* (2012)

slide 3-11 (35%)

LINKED LIST STACK QUEUE DLL DEQUE

22 (head) → 2 → 77 → 6 → 43 → 76 → 55 (6/vtx) → 89 (tail)

3-11. Insert(i, v) - In Between,  $i \in [1..N-1]$

With the Linked List traversal `Get(i)` sub-routine, we can now implement insertion in the middle of the Linked List as follows (in C++):

```
Vertex* pre = Get(i-1); // traverse to (i-1)-th vertex, O(N)
aft = pre->next; // aft cannot be null, think about it
Vertex* vtx = new Vertex(); // create new vertex
vtx->item = v;
vtx->next = aft; // link this
pre->next = vtx; // and this
```

Try `Insert(3, 44)` on the example Linked List [22 (head)->2->77->6->43->76->89 (tail)].

Also try `Insert(6, 55)` on the same example Linked List. This is a corner case: Insert at the position of tail item, shifting the tail to one position to its right.

This operation is slow,  $O(N)$ , due to the need for traversing the list (e.g. if  $i$  close to  $N-1$ ).

Lecture Example (auto play until done)  
Insert 55 at index 6

```
pre = head
for (int k = 0; k < i-1; k++)
    pre = pre->next
aft = pre->next
vtx = new Vertex(v)
vtx->next = aft
pre->next = vtx
```

slow — fast

About Team Terms of use

# Trabalhos correlatos

- Visualgo – Halim *et al.* (2012)
  - A ferramenta foi aplicada para dois grupos de usuários. O primeiro grupo foi composto por estudantes que participam de programação competitiva (24 respostas). Dentre os estudantes 54,1% afirmaram que a ferramenta ajudou a entender melhor o algoritmo.
  - O segundo grupo, composto por estudantes que participam de aceleração de algoritmos e estrutura de dados (7 respostas). Dentre os estudantes, 71% responderam que a ferramenta auxiliou a entender o algoritmo.



# Trabalhos correlatos

- Starlogo TNG – Klopfer *et al.* (2009)
  - Tem como objetivo permitir alunos e professores do ensino médio modelar sistemas descentralizados através da programação baseada em agentes.
  - Permite a criação de cenários gráficos.
  - Apresenta flexibilidade para formular cenários.
  - Apresenta uma representação visual da execução.
  - A codificação é realizada em blocos.
  - Se baseia no método científico onde o usuário realiza interações com o ambiente, coleta informações e realiza novas interações que resultarão em novas conclusões.

# Trabalhos correlatos

- Starlogo TNG – Klopfer *et al.* (2009)

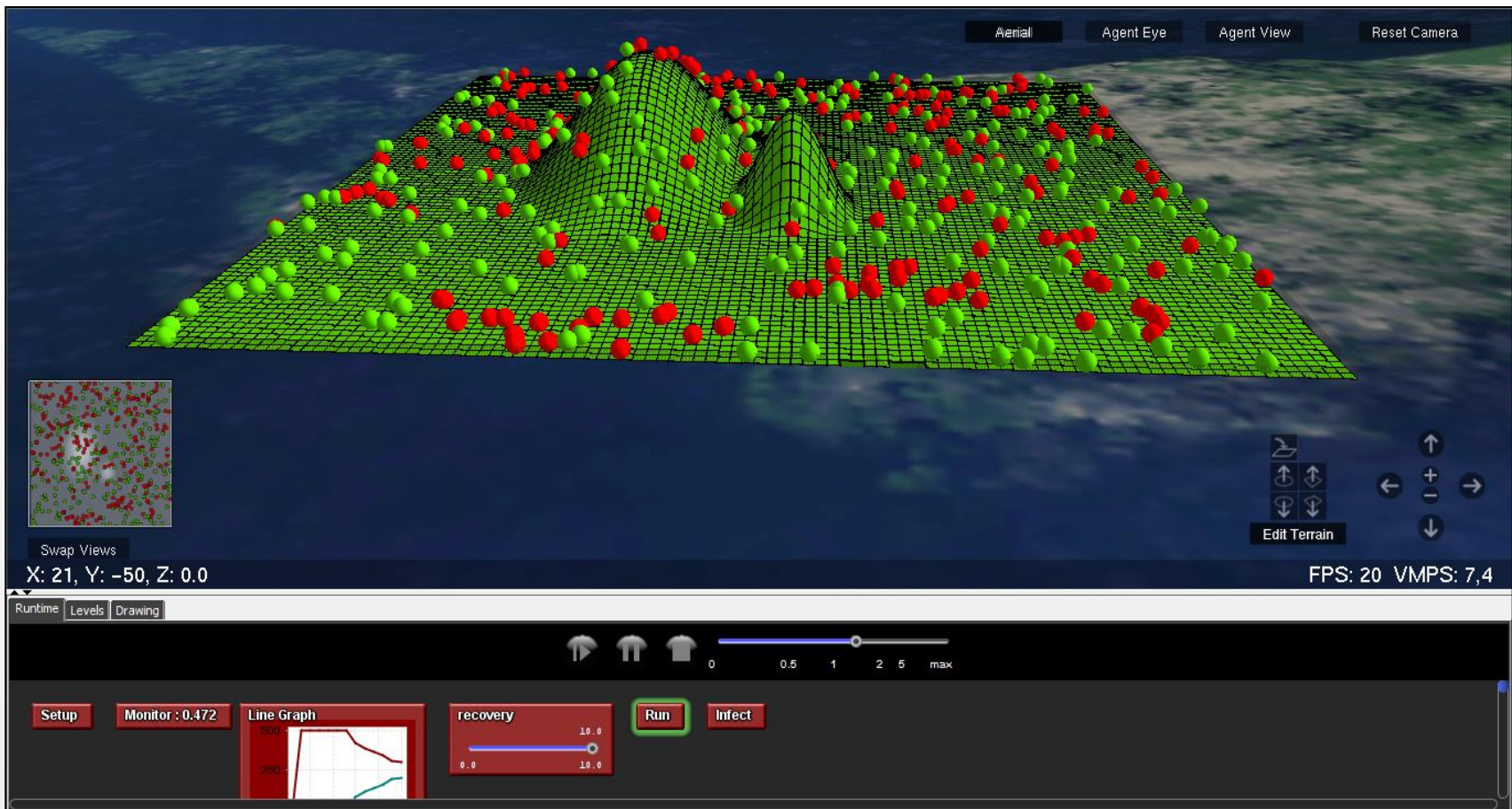
The screenshot displays the StarLogo TNG software interface, which is used for creating and running agent-based models. The interface is divided into several sections:

- Left Panel (Subsets):** A vertical menu with categories such as Setup and Run, Movement, Logic, Controls, Terrain, Traits, Text, Math, Other Agents, Procedure, Variables, List, Colors, Shapes, and Sounds.
- Top Panel:** Includes a 'Factory' button, 'Edit Breeds' button, a zoom slider set to 100%, and a search bar for blocks.
- Main Workspace:** The central area where the user constructs the model using various blocks. Visible blocks include:
  - Setup:** 'clear everyone', 'num 500', 'do', 'set color green', 'test random 100 < 10', 'if then turtles set immune 1', 'else turtles set immune 0', and 'scatter turtles'.
  - Turtles - Recover:** 'test random 100 < global recovery', 'if then set color green'.
  - Line Graph:** 'count everyone with color = green' and 'count everyone with color = red'.
  - Turtles - Immunity:** 'test turtles immune = 0', 'if then set color red'.
  - Turtles - Infection:** 'test random 100 < 10', 'if then set color red', 'Setup'.
  - Collision:** 'test color of ID collidee = red', 'if then turtles immunity'.
- Right Panel (Monitors):** A small window showing 'Turtles Setup' and 'Global Runtime Coll Monitor'.
- Bottom Panel:** A 'Setup' button and a trash can icon.

A yellow tooltip box is visible, stating: "Through interactions with others, the infected can spread the disease."

# Trabalhos correlatos

- Starlogo TNG – Klopfer *et al.* (2009)



# Trabalhos correlatos

- Starlogo TNG – Klopfer *et al.* (2009)
  - Concluiu que o uso de jogos, simulações e programação proporcionam o potencial de promover o entendimento do estudante.

# Trabalhos correlatos

- Furbot Web – Kopsch (2016)
  - Tem como objetivo desenvolver uma plataforma web adaptativa para ensino de programação, tendo como base um framework para representações visuais e aprendizado lógico, chamado FURBOT.
  - Permite o cadastro de turmas.
  - Permite o gerenciamento de exercícios com suporte a correção do professor.
  - Possui o gerenciamento de perguntas.
  - Apresentação uma representação visual da execução do meio do framework FURBOT.
  - A programação é realizada por meio de blocos.



# Trabalhos correlatos

- Furbot Web – Kopsch (2016)
  - O objetivo de criação de uma plataforma Web adaptativa foi realizado com sucesso.
  - A disponibilidade web facilitou a adesão de usuários e simplificou a experiência.
  - A programação em blocos é atrativa para estudantes que ainda não trabalharam com programação escrita, mas pode deixar de ser interessante para aqueles que já tiveram contato com programação escrita.

# Visão conceitual da ferramenta

- A ferramenta tem como objetivo ser uma plataforma educacional para aprendizagem de programação através da flexibilização do ensino e da visualização de algoritmos.
- Não se busca a substituição de uma figura educadora.
- A geração do conteúdo se dá principalmente através de **problemas**.
- Existem dois tipos de papéis: o educador e o estudante.
- Para facilitar o gerenciamento dos problemas, foi realizada uma integração com o Github.





# Visão conceitual da ferramenta

- Duas ações macros norteiam a ferramenta: a criação de um problema e a execução de um problema.
- A seguir será detalhado o processo de criação de um problema.

# Visão conceitual da ferramenta

- O primeiro passo consiste na definição de um título e uma descrição.
- A descrição disponibiliza um editor rico, para que o professor consiga adicionar seus próprios materiais, como imagens, links e vídeos.

# Visão conceitual da ferramenta

The screenshot displays the Smalg Platform interface. At the top left, the logo "Smalg Platform" is visible. On the top right, the user's name "Adriener Andrade" is shown next to a profile picture. A left sidebar menu contains the following items: "Problemas", "Novo problema", "Editar problema", "Executar problema", "Selecionar", "Executar", "Auxiliares", and "Ajuda". The main content area features a vertical progress indicator on the left with four steps: 1. Descrição, 2. Contrato, 3. Cenários, and 4. Solução. The "Descrição" step is currently active. It includes a text input field for the title with the placeholder "Defina um título para o seu problema" and a value of "Título I". Below this is a rich text editor for the description with the placeholder "Defina a descrição" and "Insert text here ...". The rich text editor toolbar includes options for bold, italic, underline, link, unlink, quote, code, heading (H1, H2), list, ordered list, subscript, superscript, indent, outdent, text color, background color, font family (Sans Serif), text size, strikethrough, undo, redo, and insert link. At the bottom of the description section are two buttons: "ANTERIOR" and "PRÓXIMO". An "AJUDA" button is located in the top right corner of the description section.

# Visão conceitual da ferramenta

- O segundo passo consiste na definição de uma interface, que se refere a abstração da implementação. O termo utilizado na ferramenta para esse conceito foi **contrato**.
- No contrato é definido o nome, os campos e os métodos que deverão ser implementados.
- A partir do contrato é gerado o esqueleto da classe de implementação da solução do problema.

# Visão conceitual da ferramenta

The screenshot displays the Smalg Platform interface. On the left, a sidebar contains navigation options: 'Problemas' (with 'Novo problema', 'Editar problema', and 'Executar problema'), and 'Auxiliares' (with 'Ajuda'). The main area shows a configuration screen for a problem, with a progress indicator on the left showing four steps: 'Descrição' (checked), 'Contrato', 'Cenários', and 'Solução'. The 'Descrição' step is active, showing a text input field for 'Nome' (containing 'T') and a table for 'Defina os campos (?)' with columns 'Nome' and 'Descrição (?)'. Below this is another table for 'Defina os métodos' with columns 'Nome', 'Parâmetros (?)', and 'Descrição (?)'. Navigation buttons 'ANTERIOR' and 'PRÓXIMO' are at the bottom. A user profile 'Adriener Andrade' is visible in the top right.

Smalg Platform

Adriener Andrade

Problemas

- Novo problema
- Editar problema
- Executar problema
- Selecionar
- Executar

Auxiliares

- Ajuda

Descrição

Defina o nome do contrato

Nome T

AJUDA

Defina os campos (?)

Nome	Descrição (?)	
		+
Nenhum campo configurado		

Defina os métodos

Nome	Parâmetros (?)	Descrição (?)	
			+
Nenhum método configurado			

ANTERIOR PRÓXIMO

Contrato

Cenários

Solução

# Visão conceitual da ferramenta

- O terceiro passo consiste na definição dos **cenários**. A ideia de um cenário é que ele funcione como pequenos trechos de execução, segmentados em pequenas partes que juntos validem o todo.
- A ideia de segmentar é que o estudante trabalhe com unidades lógicas menores, tornando a visualização e execução simplificada.
- A variável disponibilizada no cenário contendo a abstração da solução será o nome do contrato em *camel case*.
- A linguagem a ser utilizada na codificação é JavaScript.

# Visão conceitual da ferramenta

The screenshot displays the Smalg Platform interface. At the top left, the text "Smalg Platform" is visible. In the top right corner, there is a user profile icon and the name "Adriener Andrade".

The main interface is divided into two sections:

- Problemas (Problems):** A vertical sidebar on the left contains the following options:
  - Novo problema (New problem)
  - Editar problema (Edit problem)
  - Executar problema (Execute problem) - with a dropdown arrow
  - Selecionar (Select)
  - Executar (Execute)
- Auxiliares (Auxiliary):** A section below the sidebar containing:
  - Ajuda (Help)

The central workspace shows a workflow for solving a problem, indicated by a vertical list of steps on the left:

1. Descrição (Description) - marked with a blue checkmark
2. Contrato (Contract) - marked with a blue checkmark
3. Cenários (Scenarios) - marked with a blue circle containing the number 3
4. Solução (Solution) - marked with a blue circle containing the number 4

Buttons for scenario management are located in the top right of the workspace:

- ADICIONAR CENÁRIO (Add scenario)
- REMOVER CENÁRIO (Remove scenario)
- ANTERIOR (Previous)
- PRÓXIMO (Next)

An "AJUDA" (Help) button is also present in the top right corner of the workspace.



# Visão conceitual da ferramenta

- Na criação de um cenário é disponibilizado um objeto chamado `context`.

Método	Descrição
<code>context.newObject(): SmalgObject;</code>	Cria um objeto.
<code>context.newContainer(t: int): SmalgContainer;</code>	Cria um container, sendo t o tamanho do container.
<code>context.getObjects();</code>	Obtém todos os objetos criados.
<code>context.getContainers();</code>	Obtém todos os containers criados.
<code>context.getPrimitives();</code>	Obtém todas as primitivas criadas.
<code>context.clear(e: any);</code>	Limpa um elemento do contexto, sendo e o elemento.

# Visão conceitual da ferramenta

- Um objeto será representado por uma estrutura chave/valor `SmalgObject`.

Método	Descrição
<code>object.set(c: string, v: any): void</code>	Atribui um valor a partir de uma chave.
<code>object.get(c: string): any</code>	Cria um container, sendo <code>t</code> o tamanho do container.

- Um container será representado por uma estrutura semelhante a um vetor, com tamanho fixo, chamada `SmalgContainer`.

Método	Descrição
<code>container.set(i: int, v: any)</code>	Atribui um valor a partir de uma chave.
<code>container.get(c: int): any</code>	Cria um container, sendo <code>t</code> o tamanho do container.
<code>container.size(): int</code>	Retorna o tamanho do container.

# Visão conceitual da ferramenta

- Na criação de um cenário também é disponibilizado um objeto chamado `assertion`.

Método	Descrição
<code>assertion.assertEquals(e1: any, e2: any, m: string);</code>	Verifica se dois elementos são iguais, sendo v1 o primeiro elemento, v2 o segundo elemento e m a mensagem de erro.
<code>assertion.assertTrue(e: any, m: string);</code>	Verifica se o elemento passado é verdadeiro, sendo e o elemento e m a mensagem de erro.
<code>assertion.assertFalse(e: any, m: string);</code>	Verifica se o elemento passado é falso, sendo e o elemento e m a mensagem de erro.
<code>assertion.fail(m: string);</code>	Força o erro na execução, sendo m a mensagem de erro.

# Visão conceitual da ferramenta

- Exemplo de um código válido para um cenário.

```
1  listaDinamica.inicializar(2);
2
3  listaDinamica.adicionar(1);
4  listaDinamica.adicionar(2);
5  listaDinamica.adicionar(3);
6  context.clear(context.getContainers()[0]);
7  listaDinamica.adicionar(4);
8  listaDinamica.adicionar(5);
9  context.clear(context.getContainers()[0]);
10 listaDinamica.adicionar(6);
11
12 const objects = context.getObjects();
13 const containers = context.getContainers();
14 const primitives = context.getPrimitives();
15
16 assertion.assertEquals(0, objects.length, 'Não podem ser utilizados containers nesse problema.');
```

---

```
17 assertion.assertEquals(8, containers[0].properties.size, 'O tamanho do segundo container deveria ser 6.');
```

---

```
18 assertion.assertEquals(1, containers[0].container[0], 'O primeiro elemento deveria ser 1.');
```

---

```
19 assertion.assertEquals(2, containers[0].container[1], 'O segundo elemento deveria ser 2.');
```

---

```
20 assertion.assertEquals(3, containers[0].container[2], 'O terceiro elemento deveria ser 3.');
```

---

```
21 assertion.assertEquals(4, containers[0].container[3], 'O quarto elemento deveria ser 4.');
```

# Visão conceitual da ferramenta

- O quarto passo é a definição da solução. Nesta etapa é onde os cenários criados poderão ser validados e visualizados em cima de uma solução.
- A classe para implementação gerada para ser utilizada na solução é apenas um esqueleto contendo o nome, os campos e os métodos definidos no contrato, conforme a sua documentação.

# Visão conceitual da ferramenta

The screenshot displays the Smalg Platform interface. At the top left, the text "Smalg Platform" is visible next to a hamburger menu icon. In the top right corner, there is a user profile icon and the name "Adriener Andrade".

The main content area is divided into a left sidebar and a central workspace. The sidebar, titled "Problemas", contains the following options: "Novo problema" (highlighted in blue), "Editar problema", "Executar problema" (with a dropdown arrow), "Selecionar", "Executar", and "Ajuda". Below this, under the heading "Auxiliares", there is an "Ajuda" option with a question mark icon.

The central workspace features a vertical progress indicator on the left with four steps: "Descrição" (checked), "Contrato" (checked), "Cenários" (checked), and "4 Solução" (not checked). The "Descrição" step is active, showing the text "Selecione um cenário para executar" above a dropdown menu. A mouse cursor is hovering over the dropdown. In the top right of the workspace, there is an "AJUDA" button. At the bottom of the workspace, there are two buttons: "ANTERIOR" and "CRIAR PROBLEMA".

# Visão conceitual da ferramenta

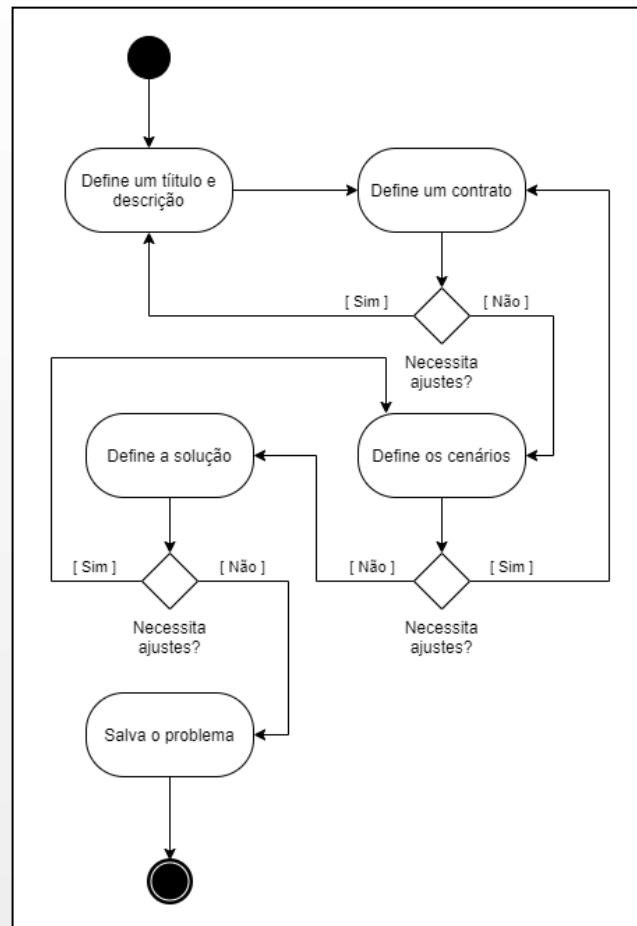
The screenshot displays the Smalg Platform interface. On the left, a sidebar contains navigation options: 'Problemas' (with sub-options 'Novo problema', 'Editar problema', 'Executar problema'), 'Auxiliares', and 'Ajuda'. The main area is titled 'Smalg Platform' and features a user profile 'Adriener Andrade' in the top right. A vertical progress indicator on the left shows four steps: 'Descrição', 'Contrato', 'Cenários', and '4 Solução'. The 'Descrição' step is active, showing a code editor with the following JavaScript code:

```
70
71
72 * Retorna o elemento presente na posição informada
73 */
74 obter(posicao) {
75   let noAtual = this.noInicial;
76   for (let i = 0; i < posicao; i++) {
77     noAtual = noAtual.get('proximo');
78   }
79   return noAtual.get('valor');
80 }
81 /**
82 * Retorna o tamanho da lista
83 */
84 obterTamanho() {
85   return this.tamanho;
86 }
```

Buttons for 'EXECUTAR', 'CENTRALIZAR', 'Mudar velocidade', 'ADICIONAR', 'DESCRIÇÃO DO CENÁRIO', 'SOLUÇÃO', 'ANTERIOR', and 'CRIAR PROBLEMA' are visible. An 'AJUDA' button is also present in the top right of the main area.

# Visão conceitual da ferramenta

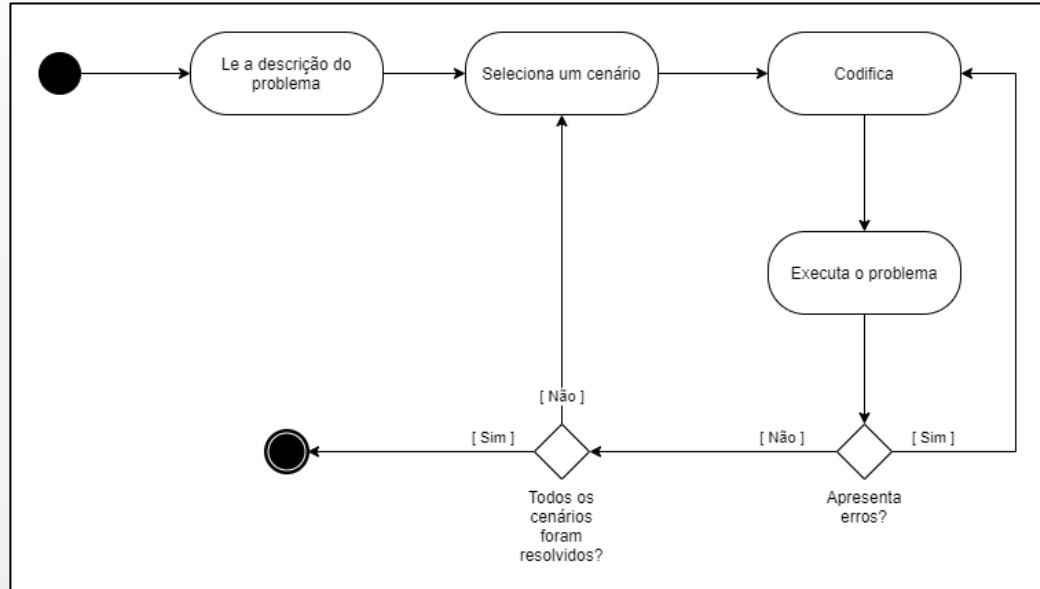
- No geral, o processo pode ser descrito pelo seguinte fluxograma.





# Visão conceitual da ferramenta

- Após a criação de um problema, ele poderá ser executado.
- Ao usuário abrir o problema ele terá disponibilizado a descrição cadastrada e um local no qual terá a classe abstrata que deverá ser implementada. O processo é semelhante a etapa de solução da criação de um problema.



# Visão conceitual da ferramenta

- Para a solução o estudante deverá utilizar o objeto `context` para criar objetos e vetores. Objetos e vetores do JavaScript puros, não serão considerados durante a execução.

Método	Descrição
<code>context.newObject(): SmalgObject;</code>	Cria um objeto.
<code>context.newContainer(t: int): SmalgContainer;</code>	Cria um container, sendo t o tamanho do container.

# Visão conceitual da ferramenta

The screenshot shows the Smalg Platform interface for creating a dynamic list (Array List). The interface is divided into several sections:

- Problemas (Problems):** Located on the left sidebar, it includes options like "Novo problema", "Editar problema", "Executar problema", "Selecionar", and "Executar".
- Descrição (Description):** The main content area, containing a detailed explanation of dynamic lists and their characteristics. Callout 1 points to the title, and callout 2 points to the description text.
- Código (Code):** The code editor area, showing a JavaScript function for adding an element to a list. Callout 3 points to the "Executar" button in the sidebar, callout 4 to the "Adicionar" dropdown, callout 5 to the "DESCRÇÃO DO CENÁRIO" button, callout 6 to the "SOLUÇÃO" button, callout 7 to the code editor, callout 8 to the "CENTRALIZAR" button, callout 9 to the "AJUDA" button, callout 10 to the "EXECUTAR" button, callout 11 to the code editor, callout 12 to the "CENTRALIZAR" button, callout 13 to the "5x" dropdown, callout 14 to the "Valores Iniciais" panel, and callout 15 to the playback controls.
- Valores Iniciais (Initial Values):** A panel on the right side of the code editor, showing a vertical list of green circles representing initial values.
- Playback Controls:** A set of buttons at the bottom center of the code editor, including play, pause, and stop buttons.

# Visão conceitual da ferramenta

The screenshot displays the Smalg Platform interface. At the top left, the logo and name 'Smalg Platform' are visible. On the top right, there is a user profile icon for 'Adriener Andrade'. A left sidebar contains navigation options: 'Problemas', 'Novo problema', 'Editar problema', 'Executar problema', 'Selecionar', 'Executar', 'Auxiliares', and 'Ajuda'. The main content area is divided into two sections. The upper section contains a text description of a linked list problem in Portuguese, explaining that a singly linked list is not efficient for direct access to a specific position because it requires traversing all previous nodes. It mentions variations like doubly linked lists and states that the exercise will focus on a simple singly linked list. The lower section is titled 'Código' and includes a 'Remover' dropdown menu, a 'DESCRIÇÃO DO CENÁRIO' button, a 'SOLUÇÃO' button, and icons for code editing. Below these are 'EXECUTAR' and 'CENTRALIZAR 3x' buttons. The code editor shows a JavaScript function named 'remover' that takes an element to be removed as an argument. It initializes 'noAnterior' to null and 'atual' to 'this.noInicial'. A 'while' loop iterates through the list until it finds the element to be removed. If found, it updates the 'proximo' pointer of the previous node to skip the current node and decrements the list size. If not found, it returns. Comments at the bottom indicate the function's purpose: to verify if an element exists in the list.

Smalg Platform

Adriener Andrade

Problemas

Novo problema

Editar problema

Executar problema

Selecionar

Executar

Auxiliares

Ajuda

Performática para operações de adição e exclusão de elementos, a lista encadeada não se torna performática em operações de acesso direto a uma posição da lista, pois precisa percorrer todos os nós anteriores.

Existem variações da lista encadeada, como a lista duplamente encadeada, na qual cada nó aponta para o seu sucessor e o seu antecessor.

Neste exercício trabalharemos apenas com a lista encadeada simples.

Você deverá implementar o método de adição, remoção, obtenção de um elemento, tamanho da lista e verificação se a lista contém um determinado elemento.

Código

Selecione um cenário para executar

Remover

DESCRIÇÃO DO CENÁRIO

SOLUÇÃO

EXECUTAR

CENTRALIZAR 3x

```
38  remover(elemento) {
39      let noAnterior = null;
40      let atual = this.noInicial;
41      while (atual) {
42          const valor = atual.get('valor');
43          if (valor === elemento) {
44              noAnterior.set('proximo', atual.get('proximo'));
45              atual.set('proximo', null);
46              this.tamanho--;
47              return;
48          } else {
49              noAnterior = atual;
50              atual = atual.get('proximo');
51          }
52      }
53      /**
54       * Verifica se um elemento existe na lista
55       */
56  }
```

# Visão conceitual da ferramenta

Smalg Platform Adriener Andrade

Problemas

- Novo problema
- Editar problema
- Executar problema
- Selecionar
- Executar

Auxiliares

- Ajuda

## Criando uma Lista Encadeada

Descrição

Uma Lista Encadeada é uma estrutura de dados linear composta por nós que apontam para o seu próximo vizinho.

Para criar uma lista encadeada, basta definir um nó inicial e para cada nó existente salvar o seu sucessor.

A imagem abaixo ilustra uma lista encadeada:

```
graph LR; N1[250] --> N2[15]; N2 --> N3[45]; N3 --> N4[189];
```

Perfomática para operações de adição e exclusão de elementos, a lista encadeada não se torna performática em operações de acesso direto a uma posição da lista, pois precisa percorrer todos os nós anteriores.

Existem variações da lista encadeada, como a lista duplamente encadeada, na qual cada nó aponta para o seu sucessor e o seu antecessor.

Neste exercício trabalharemos apenas com a lista encadeada simples.

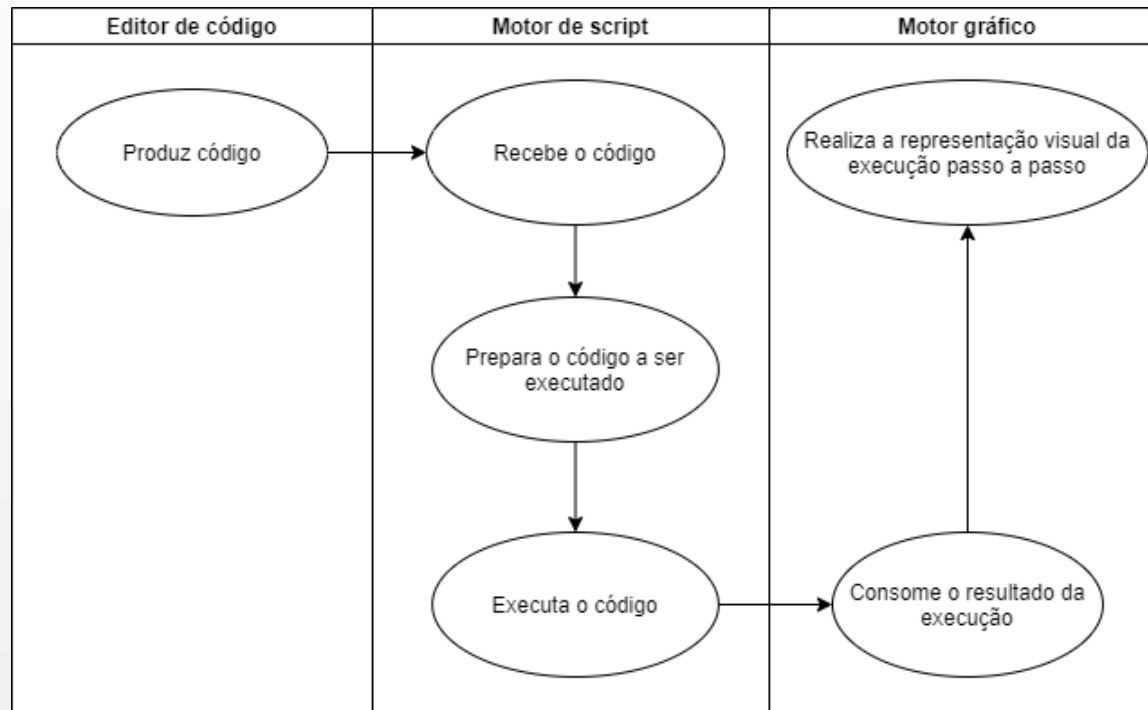
Você deverá implementar o método de adição, remoção, obtenção de um elemento, tamanho da lista e verificação se a lista contém um determinado elemento.

# Implementação

- A ferramenta foi desenvolvida para a plataforma web para os navegadores Microsoft Edge v87.0.664.41, Google Chrome v87.0.4280.66 e Mozilla Firefox v83.0, não tendo foco em disponibilidade para plataformas móveis.
- Como alternativa para facilitar o uso, foi implementada uma integração com o Github através de autenticação OAuth2.0.
- Para desenvolvimento do frontend foi utilizado o framework Angular 10+, tendo como base o projeto Ngx Admin.
- O componente utilizado para realizar a codificação do cenário e da solução do problema foi o Monaco Editor.
- Para a representação visual foi utilizado a biblioteca Cytoscape.js.
- Como opção de editor de texto rico foi utilizado a biblioteca QuillJS.

# Implementação

- O processo de execução de um problema foi dividido em três componentes principais: o editor de código, o motor de script e o motor gráfico.

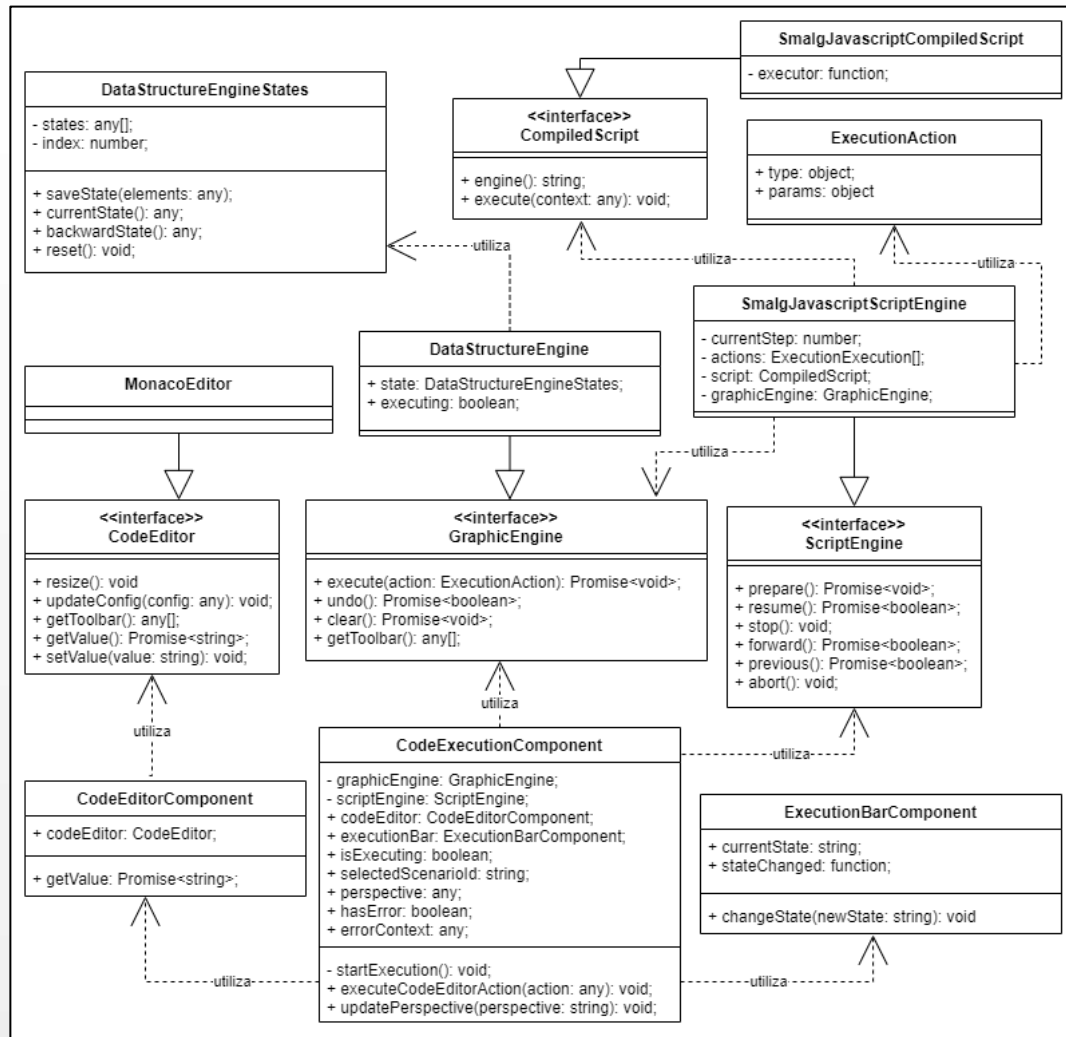


# Implementação

- Para cada um dos componentes principais apresentados anteriormente foram criadas interfaces (CodeEditor, ScriptEngine e GraphicEngine), sendo as suas implementações disponibilizadas por um provider.



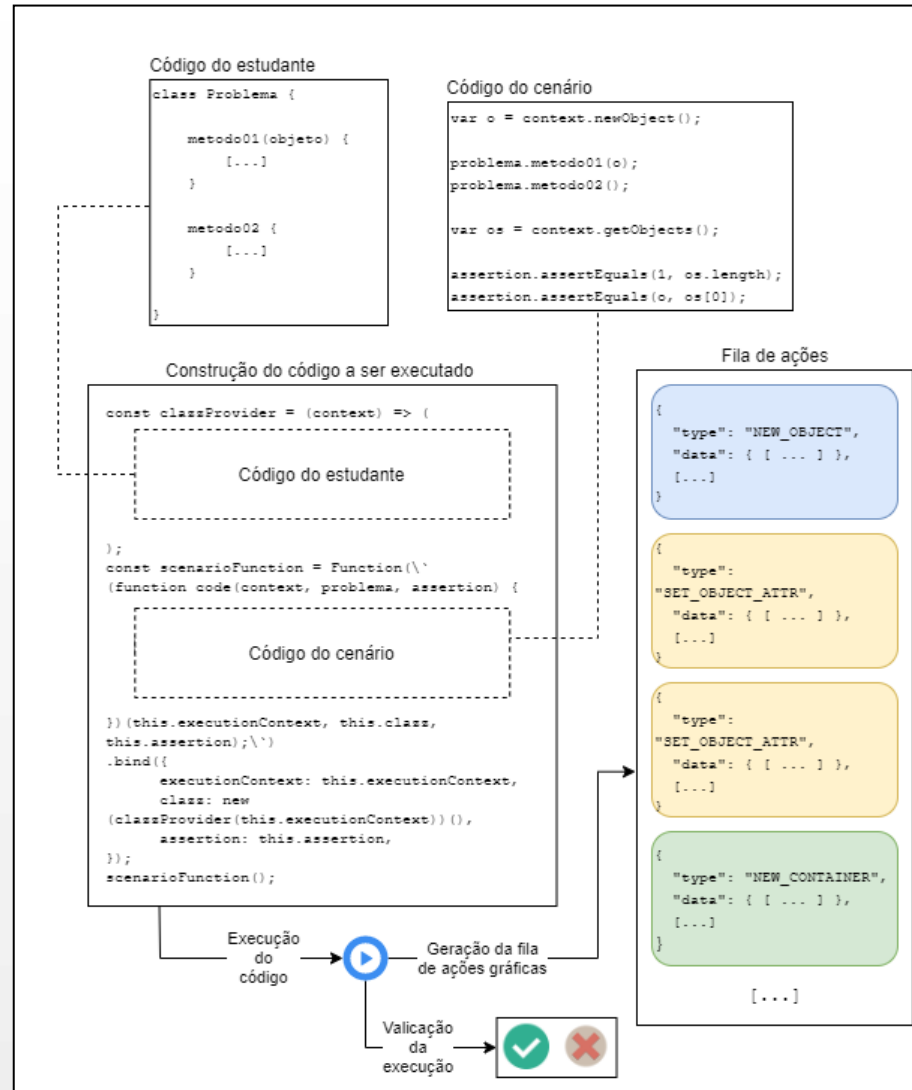
# Implementação



# Implementação

- O motor de script trabalha preparando a função que será executada.
- Ele trabalha com o construtor `Function` do JavaScript, se aproveitando do interpretador nativo da linguagem.
- Para injeção do contexto de execução (`context`) e do objeto de assertivas (`assertion`), foram utilizados os métodos `bind` e `apply` nativos do JavaScript.

# Implementação



# Implementação

```
export class SmalgJavascriptContext {  
  
  private elements: { [key: string]: SmalgType } = {};  
  
  constructor(private actions: ExecutionAction[]) {}  
  
  newObject() {  
    return this.createNewElement(() => new SmalgObject(this.actions));  
  }  
  
  newContainer(size: number) {  
    return this.createNewElement(() => new SmalgContainer({ size }, this.actions));  
  }  
  
  newPrimitive(value: string | number | boolean) {  
    return this.createNewElement(() => new SmalgPrimitive(value, this.actions));  
  }  
  
  clear(elems: SmalgType | SmalgType[]) {  
    if (!elems) {  
      return;  
    }  
    if (Array.isArray(elems)) {  
      elems.forEach(elem => this.clearElement(elem));  
    } else {  
      this.clearElement(elems);  
    }  
  }  
}
```

# Implementação

```
set(index: number, value?: SmalgType | string | boolean | number) {  
  this.validateIndex(index);  
  
  if (isPrimitive(value)) {  
    value = new SmalgPrimitive(value, this.actions);  
  }  
  
  value = value?.__reference__();  
  this.actions.push({  
    type: DataStructureAction.SET_CONTAINER_SLOT,  
    params: { id: this.__getId__(), index, value: value?.__getId__() },  
  });  
  this.container[index] = value;  
}  
  
get(index: number) {  
  this.validateIndex(index);  
  
  const value = (this.container[index])?.__reference__();  
  const params = { id: this.__getId__(), index, value: value?.__getId__() };  
  this.actions.push({ type: DataStructureAction.GET_CONTAINER_SLOT, params });  
  
  return value?.__value__();  
}
```

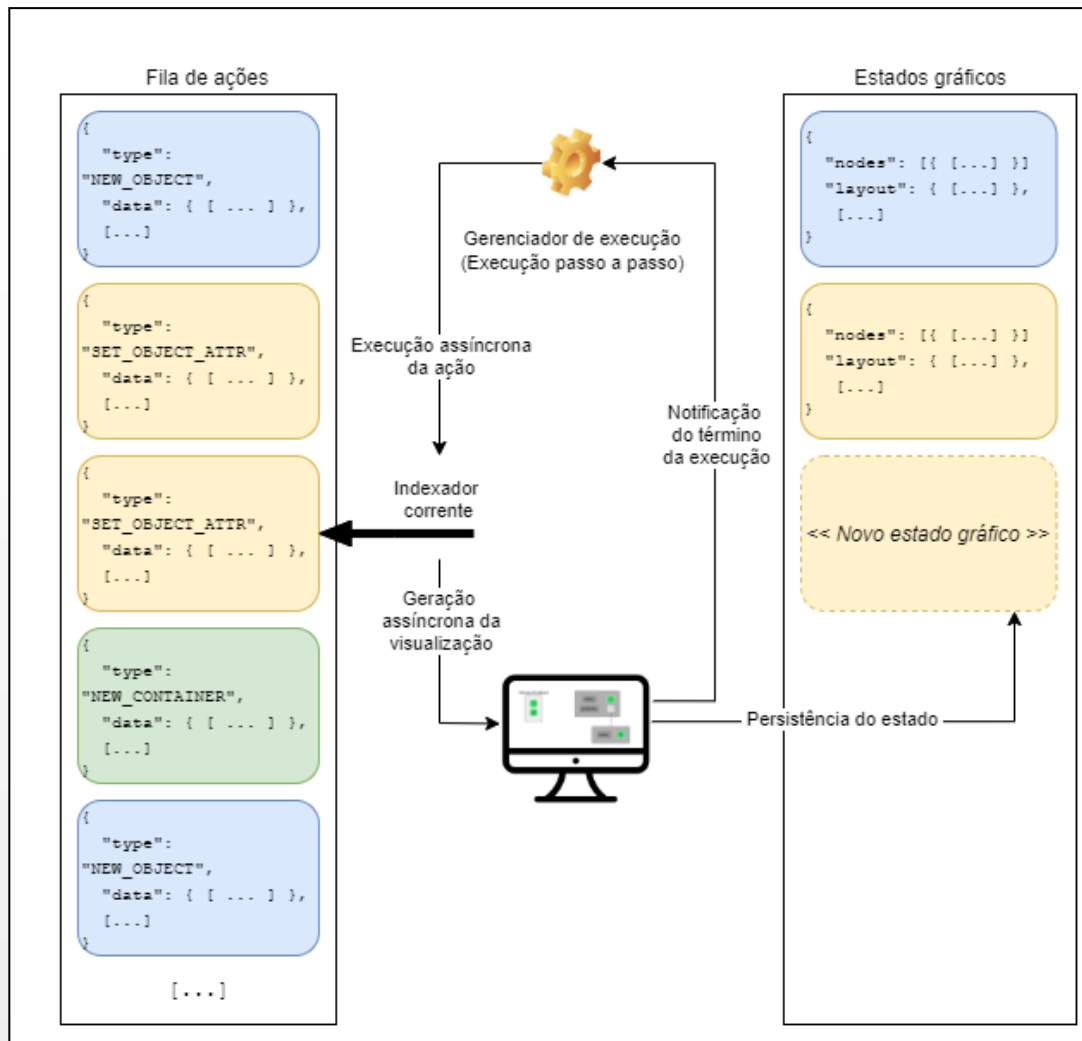
# Implementação

- Ao todo foram criadas as seguintes ações:
  - CREATE\_CONTAINER;
  - CREATE\_OBJECT;
  - CREATE\_PRIMITIVE;
  - DELETE\_ELEMENT;
  - GET\_CONTAINER\_SLOT;
  - GET\_OBJECT\_SLOT;
  - SET\_CONTAINER\_SLOT;
  - SET\_OBJ\_ATTR.

# Implementação

- O motor gráfico trabalha renderizando visualmente cada ação.
- Durante a execução da visualização existe um gerenciador de execução, que trata os comandos de pausar, retroceder, próximo passo, resumir ou parar a visualização.

# Implementação





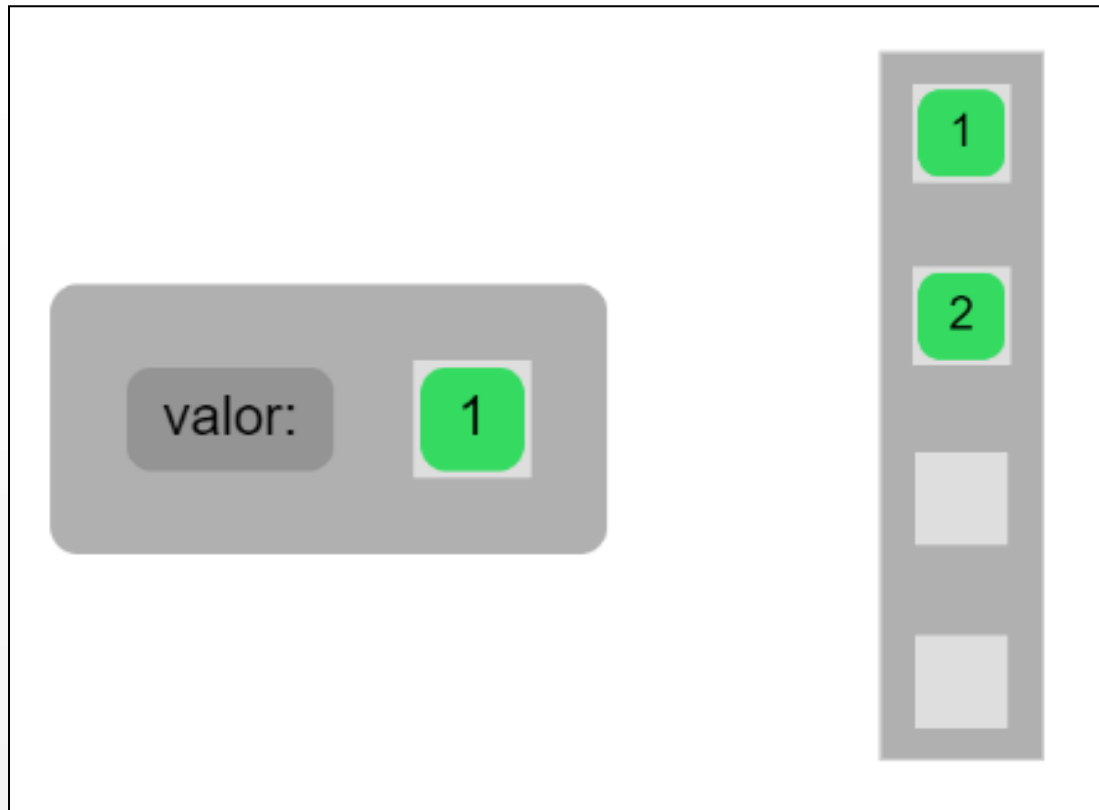
# Implementação

```
7 export class CreateContainerAction implements CytoscapeActionHandler {
8
9   private layoutHandler: ContainerLayoutHandler = new ContainerLayoutHandler();
10
11  async handle(cytoscape: any, action: ExecutionAction): Promise<void> {
12    const { id, size } = action.params;
13    const containerElement = await $add(cytoscape, {
14      data: {
15        id,
16        type: ElementTypes.CONTAINER,
17      },
18    });
19    await this.layoutHandler.setInitialPosition(containerElement);
20    for (let i = 0; i < size; i++) {
21      const slotElement = await $add(cytoscape, {
22        data: [
23          {
24            id: `${id}_${i}`,
25            type: 'container_slot',
26            index: i,
27          },
28        ],
29        classes: ['slot'],
30      });
31      await this.layoutHandler.moveToContainer(containerElement, slotElement);
32      slotElement.move({ parent: id });
33    }
34    await this.layoutHandler.run(containerElement);
35  }
36
37  name(): string {
38    return DataStructureAction.CREATE_CONTAINER.name;
39  }
40 }
41
```

# Implementação

```
export class SetContainerSlotAction implements CytoscapeActionHandler {  
  
  private layoutHandler: ContainerLayoutHandler = new ContainerLayoutHandler();  
  
  async handle(cytoscape: any, action: ExecutionAction): Promise<void> {  
    const id: string = action.params.id;  
    const index: number = action.params.index;  
    const value: string = action.params.value;  
    const valueElement = value ? $id(cytoscape, value) : null;  
    const slotElement = $id(cytoscape, `${id}_${index}`);  
  
    this.clearCurrentValue(cytoscape, slotElement);  
    if (valueElement) {  
      await this.setValue(cytoscape, slotElement, valueElement);  
    }  
  
    await this.layoutHandler.run($id(cytoscape, id));  
  }  
  
  private clearCurrentValue(cytoscape, slotElement): void {  
    const slotRelations = slotElement.neighborhood();  
    if (slotRelations.length > 0) {  
      $removeRelation(cytoscape, slotElement, slotRelations[0]);  
    }  
    const currentValueElement = slotElement.children()[0];  
    if (currentValueElement) {  
      cytoscape.remove(currentValueElement);  
    }  
  }  
  
  private async setValue(cytoscape, slotElement, valueElement) {  
    if (valueElement.data('type') === ElementTypes.PRIMITIVE) {  
      await this.layoutHandler.moveToSlot(slotElement, valueElement);  
      valueElement.move({parent: slotElement.id()});  
    } else {  
      $addRelation(cytoscape, slotElement, valueElement);  
    }  
  }  
  
  name() {  
    return DataStructureAction.SET_CONTAINER_SLOT.name;  
  }  
}
```

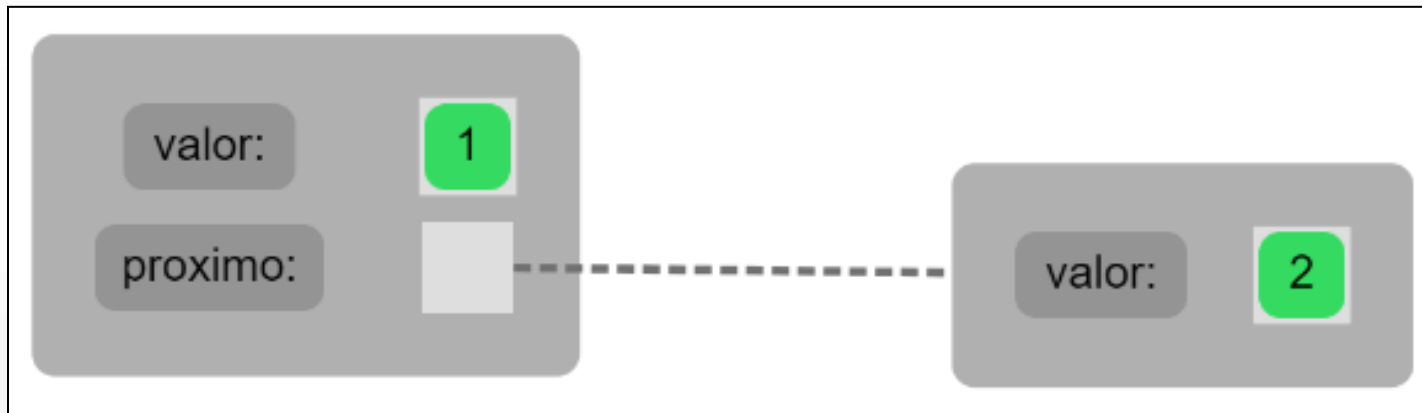
# Implementação



# Implementação



# Implementação



# Resultados e discussões

- Foram realizados 3 testes distintos com diferentes usuários.
- O primeiro foi um teste de usabilidade com um usuário externo, com conhecimento prévio em programação, cursando Ciência da Computação na Universidade Unisul.
- O segundo foi uma oficina realizada com quatro alunos cursando o segundo semestre de Ciência da Computação na FURB.
- O terceiro foi uma chamada realizada com dois professores de Ciência da Computação da FURB, um responsável pela disciplina de Introdução à Programação e outro relacionado a disciplina de Algoritmo e Estrutura de Dados.

# Resultados e discussões

- O teste de usabilidade foi realizado através de uma chamada, sendo as ações do usuário monitoradas pelo autor.
- Teve como principal objetivo encontrar erros, diferentes caminhos de uso e preparar a aplicação para os próximos dois testes a serem realizados.

# Resultados e discussões

- A oficina foi aplicada com os alunos, de maneira que o autor do trabalho assumiu o papel do professor.
- Foram realizadas em média em 1h, aplicando algoritmos de Bubble Sort, Lista Encadeada e Lista Dinâmica (Array List). Todos os problemas estão disponíveis na documentação e foram desenvolvidos pelo autor.
- Aplicado um questionário para ser respondido após o uso da ferramenta.



# Resultados e discussões

Pergunta	Respostas disponíveis
Tendo em vista o problema Bubble Sort, você considera que a ferramenta te auxiliou a entender o conceito envolvido?	Sim, muito. – 75% (3/4). Sim, um pouco. – 25% (1/4). Não, pois já compreendia. – 0% (0/4). Não, continuo sem compreender. – 0% (0/4)
Tendo em vista o problema da lista encadeada, você considera que a ferramenta te auxiliou a entender o conceito envolvido?	Sim, muito. – 75% (3/4). Sim, um pouco. – 25% (1/4). Não, pois já compreendia. – 0% (0/4). Não, continuo sem compreender. – 0% (0/4)
Tendo em vista o problema lista dinâmica (Array List), você considera que a ferramenta te auxiliou a entender o conceito envolvido?	Sim, muito. – 25% (1/4). Sim, um pouco. – 75% (3/4). Não, pois já compreendia. – 0% (0/4). Não, continuo sem compreender. – 0% (0/4)
Como você avalia o grau de dificuldade para utilizar a ferramenta?	Não tive dificuldades. – 25% (1/4). Baixo. – 75% (3/4). Médio. – 0% (0/4). Alto. – 0% (0/4). Muito alto. – 0% (0/4).
Como você descreveria sua curva de aprendizado para utilizar a ferramenta?	Curta, um dia. – 100% (4/4). Média, três dias. – 0% (0/4). Longa, mais de três dias. – 0% (0/4). Não sei dizer. – 0% (0/4).
Como você avalia a documentação disponibilizada?	Muito boa. – 25% (1/4). Boa. – 50% (2/4). Regular. – 25% (1/4). Ruim. – 0% (0/4). Péssima. – 0% (0/4). Não cheguei a ler. – 0% (0/4).
No geral, acredito que a ferramenta:	Foi útil para o meu aprendizado. – 100% (4/4). Não contribuiu em nada para o meu aprendizado. – 0% (0/4).

# Resultados e discussões

- Realizada uma chamada com o professor apresentando a ferramenta e explicando o seu propósito.
- Disponibilizado um tempo aproximado de uma semana para utilização da ferramenta e levantamento de considerações.
- Aplicado um questionário para ser respondido após o uso da ferramenta.

# Resultados e discussões

Pergunta	Respostas
Como você avalia a flexibilidade para criação de problemas?	Muito boa. – 100% (2/2). Boa, porém poderia ser um pouco mais flexível. – 0% (0/2). Regular, o modelo é um pouco engessado. – 0% (0/2). Ruim, não me ofereceu muita flexibilidade. – 0% (0/2). Péssima, não ofereceu flexibilidade nenhuma. – 0% (0/2).
Como você avalia a facilidade para a criação de problemas, considerando a flexibilidade proposta?	Muito boa. – 50% (1/2). Boa, porém poderia ser um pouco mais fácil. – 0% (0/2). Regular, não tão fácil nem tão complexo. – 50% (1/2). Ruim, é um processo um pouco complexo. – 0% (0/2). Péssima, extremamente difícil. – 0% (0/2).
Como você avalia o potencial da plataforma para ensinar de modo autodidata para os alunos?	Muito bom. – 0% (0/2). Bom, porém o aluno terá algumas dificuldades. – 50% (1/2). Regular, o aluno terá dificuldades medianas para usar. – 50% (1/2). Ruim, o aluno terá várias dificuldades para usar. – 0% (0/2). Péssimo, o aluno não terá condições de usar. – 0% (0/2).
Como você avalia o potencial da plataforma para ser utilizada como ferramenta auxiliar ao professor no ensino?	Muito bom. – 100% (2/2). Bom. – 0% (0/2). Regular. – 0% (0/2). Ruim. – 0% (0/2). Péssimo. – 0% (0/2).
Você utilizaria a ferramenta aplicando diretamente com os alunos?	Sim. Acredito que a interação dos alunos pode agregar ao aprendizado. – 50% (1/2) Não, mas montaria meus próprios problemas e apresentaria a visualização para auxiliar na explicação. – 50% (1/2). Não utilizaria a ferramenta. – 0% (0/2).

# Resultados e discussões

Pergunta	Respostas
Se você for aplicar diretamente com os alunos, como você avalia o nível de acompanhamento e instrução que deverá ser realizado por parte do professor?	Nenhum. – 0% (0/2). Quase nenhum. – 0% (0/2). Razoável. – 100% (2/2). Um acompanhamento integral. – 0% (0/2). Não utilizaria com os alunos. – 0% (0/2).
Como você classifica a visualização apresentada?	Muito boa. – 50% (1/2). Boa. O conceito pode ser entendido, mas tem alguns pontos que poderiam melhorar. – 50% (1/2). Regular. – 0% (0/2). Ruim. – 0% (0/2). Péssima. – 0% (0/2).
Como você classifica a curva de aprendizado para conseguir utilizar a ferramenta, por parte do professor?	Curta, um dia. – 50% (1/2). Média, três dias. – 50% (1/2). Relativamente longa, uma semana. – 0% (0/2). Longa, mais de uma semana. – 0% (0/2).
No geral, considero a ferramenta como uma possibilidade de ensino:	Muito boa. – 50% (1/2). Boa. – 50% (1/2). Regular. – 0% (0/2). Ruim. – 0% (0/2). Péssima. – 0% (0/2).
Como você avalia a qualidade da documentação da plataforma?	Muito boa – 100% (2/2). Boa – 0% (0/2). Regular – 0% (0/2). Ruim – 0% (0/2). Péssima – 0% (0/2).

# Comparativo com os trabalhos correlatos

- A ferramenta agregou diferentes aspectos de cada um dos trabalhos apresentados.
- A representação visual era uma característica presente em todos os correlatos.
- Do trabalho de Halim *et al.* (2012) foi incorporada a ideia do ambiente unificado, proporcionando a flexibilidade para modelar problemas e criar um único meio de linguagem de ensino.
- Em relação ao trabalho de Klopfer *et al.* (2009), foi incorporado o aspecto do aprendizado incremental, no qual o estudante produz um conteúdo, o executa e coleta os resultados dessa execução.

# Comparativo com os trabalhos correlatos

- Em relação ao trabalho de Kopsch (2016), a principal característica que se destaca é a criação de uma plataforma adaptativa. Por outro lado, o foco estava na programação escrita e não em blocos.

# Conclusões

- A ferramenta atendeu o propósito de auxiliar no processo educacional, disponibilizando a representação visual e flexibilização do ensino.
- A programação escrita pode ser dificultosa para alunos em fases muito iniciais.
- O recurso da visualização, acompanhado pela sua execução passo a passo, demonstrou ter forte contribuição no aprendizado.
- Existe uma pequena curva de aprendizado para se habituar a ferramenta.
- A documentação disponibilizada apresentou um papel essencial no impulsionamento da ferramenta.

# Sugestões de extensão

- Criação de um interpretador próprio que transpile para o modelo de linguagem atual;
- Execução passo a passo em cima do código que está sendo executado;
- Implementação de um algoritmo para organização automática dos elementos visuais.
- Implementação de um console que permita a interação do usuário em tempo de execução.



Obrigado!