Documentação do Código - Bingo de Modos Silogísticos

Projeto Spring Boot + HTML para sorteio de argumentos válidos, revelação de gabarito e detecção automática de bingo em cartelas.

Base técnica: Spring Boot 3.5.0, Java 21, Maven, Thymeleaf (apenas para servir o HTML).

1. Visão geral do sistema

O sistema implementa uma dinâmica de sala de aula: a cada jogada, a aplicação sorteia um argumento categórico válido associado a um modo silogístico tradicional (por exemplo, Barbara, Celarent). O aluno identifica o nome do modo. Quando o professor solicita, o sistema revela o gabarito (nome do modo, figura e forma). Em paralelo, o sistema registra o modo sorteado e verifica quais cartelas fecharam bingo no ciclo atual.

Regras implementadas: id neutro para evitar vazamento do modo, ciclo reiniciável via 'Novo ciclo', e destaque do termo médio nas premissas ao revelar a resposta, facilitando a identificação da figura do silogismo.

2. Estrutura de pastas

- pom.xml
- src
  - main
    - java
      - com.lorenzon.silogismos
        - SilogismosApplication.java
        - controller
          - GameController.java
        - domain
          - ArgumentCard.java
          - BingoCard.java
        - service
          - DeckService.java
          - BingoService.java
    - resources
      - application.properties
      - cards.json
      - bingo-cards.json
      - templates
        - index.html
- README.txt

3. Arquitetura em camadas

A arquitetura segue uma separação simples por responsabilidades:

- domain: modelos de dados (ArgumentCard e BingoCard).

- service: regras de negócio e estado do jogo no servidor (DeckService e BingoService).

- controller: endpoints HTTP para o front-end consumir (GameController).

- resources/templates: interface do jogo (HTML, CSS e JavaScript).

- resources/*.json: dados do baralho de argumentos e das cartelas.

4. Camada domain

4.1 ArgumentCard

Representa uma carta de argumento (um silogismo categórico válido). O id é neutro (C001, C002, ...) para não entregar o nome do modo antes do momento de revelar o gabarito.

Campo	Tipo	Significado
id	String	Identificador neutro da carta (não revela o modo).
name	String	Nome do modo (ex.: Barbara). Só aparece na resposta.
figure	int	Figura do silogismo (1 a 4).
mood	String	Forma categórica (AAA, EAE, AII etc.).
premises	List	Duas premissas categóricas.
conclusion	String	Conclusão categórica.

4.2 BingoCard

Representa uma cartela do bingo. Cada cartela possui 6 nomes de modos. Quando todos os 6 modos já foram sorteados no ciclo, a cartela fecha bingo.

Campo	Tipo	Significado
number	int	Número da cartela (1 a 50).
modes	List	Lista com 6 nomes de modos presentes na cartela.

5. Camada service

5.1 DeckService

Responsável por carregar o baralho de argumentos (cards.json), embaralhar e entregar a próxima carta. Também mantém referência para a carta atual, utilizada pela rota que revela o gabarito.

Método	Retorno	Responsabilidade
reshuffle()	void	Embaralha o baralho e reinicia o índice do baralho.
next()	ArgumentCard	Retorna a próxima carta do baralho; se chegou ao final, embaralha novamente.
current()	ArgumentCard	Retorna a carta atual (última retornada por next).

5.2 BingoService

Responsável por carregar as cartelas (bingo-cards.json) e controlar o estado do ciclo do jogo: modos já sorteados, quantidade de jogadas e lista de cartelas vencedoras (bingo).

Método	Retorno	Responsabilidade
reset()	void	Reinicia o ciclo: zera modos sorteados, vencedores e contador.
registerDraw(modeName)	void	Registra um modo sorteado e incrementa o contador de jogadas.
checkNewWinners()	List	Verifica cartelas que fecharam bingo agora e retorna apenas novos vencedores.
getDrawnModes()	Set	Retorna conjunto de modos já sorteados (somente leitura).
getWinners()	List	Retorna lista acumulada de cartelas vencedoras no ciclo.
getDraws()	int	Retorna total de jogadas no ciclo.

6. Camada controller (endpoints)

GameController expõe a página HTML e uma API simples para o front-end. As rotas retornam JSON.

Rota	HTTP	Descrição
/	GET	Entrega a página do jogo (index.html).
/api/next	GET	Sorteia a próxima carta, registra o modo e retorna o argumento e novos bingos.

/api/answer	GET	Revela o gabarito da carta atual (nome do modo, figura e forma).
/api/status	GET	Status do ciclo: total de jogadas, quantidade de modos distintos e vencedores.
/api/reset	POST	Inicia novo ciclo: embaralha baralho e zera bingos e contador.

7. Interface web (index.html)

A interface é uma página única com JavaScript para consumir a API. Pontos principais:

- Cabeçalho: botões de controle e regra de aula.

- Quadro central: premissas e conclusão em destaque, sem quebra de linha (rolagem horizontal quando necessário).

- Resposta: exibida somente após clique em 'Mostrar resposta', com destaque em vermelho.

- Destaque do termo médio: ao mostrar resposta, o JavaScript identifica o termo comum às premissas e aplica realce nas duas frases.

7.1 Destaque do termo médio (identificação da figura)

O JavaScript tenta interpretar proposições no formato: 'Todo S é P', 'Nenhum S é P', 'Algum S é P', 'Algum S não é P'. Ele extrai sujeito e predicado das duas premissas, encontra o termo comum (termo médio) e o destaca com um fundo. Em seguida, indica se o termo médio aparece como sujeito ou predicado em cada premissa.

8. Arquivos de dados (JSON)

cards.json contém o baralho de argumentos válidos. Cada carta possui duas premissas e uma conclusão, e o gabarito (nome, figura e forma). bingo-cards.json contém as 50 cartelas, cada uma com 6 nomes de modos.

9. Como executar

mvn spring-boot:run

Acessar: http://localhost:8080

10. Observações para operação em sala

Se aparecer no log do Tomcat uma mensagem sobre caracteres inválidos no método HTTP, isso costuma indicar tentativa de acesso HTTPS na porta HTTP (por exemplo, usar https://localhost:8080). Use http://localhost:8080.