

VolleyScript:

Aprendizagem ativa e interdisciplinaridade no desenvolvimento de um jogo digital

Amanda de Almeida Crispim
Instituto Federal Catarinense
Campus Araquari
amandacrispim384@gmail.com

Gustavo Henrique Trentini
Instituto Federal Catarinense
Campus Araquari
gustavotrentini22@gmail.com

Eduardo da Silva
Instituto Federal Catarinense
Campus Araquari
eduardo.silva@ifc.edu.br

Marco Antonio Torrez Rojas
Instituto Federal Catarinense
Campus Araquari
marco.rojas@ifc.edu.br

ABSTRACT

In technical courses in informatics there is a need to develop students' basic skills. The project-based approach and gamification make this development possible, it allows the development of the solution based on the perspective of a game designer, natural for digital natives. Aiming to achieve interdisciplinarity between technical and propaedeutic disciplines, these methodological approaches were used to develop a digital volleyball game. The game was developed using the object-oriented programming paradigm, for the interactive and creative interface design, the 80's indie style pixel-arts was used. The validation of the game's functionality was carried out through tests, these tests were made by volunteer students and adjustments according to their feedback. Thus, the application of active learning methodologies contributed to the success of the interdisciplinary, achieved with the development of the digital game by the students.

KEYWORDS

metodologia ativa, jogo digital, voleibol, *pixel-art*, POO.

1 INTRODUÇÃO

Este trabalho tem como objetivo aplicar os conhecimentos adquiridos e estudados durante o curso técnico em informática do Instituto Federal Catarinense (IFC) do Campus Araquari, e integrando as disciplinas de Educação Física, Matemática, Artes e Programação por meio do desenvolvimento de um jogo digital.

Visando contribuir para o desenvolvimento das habilidades básicas necessárias para os alunos e futuros profissionais da área das TICs, as metodologias ativas são utilizadas neste trabalho, principalmente a abordagem de aprendizagem baseada em projetos [1]. Também a abordagem da gamificação ou ludificação será utilizada visando prover a abordagem para se observar o problema

ou contexto seguindo a percepção de um projetista de jogos [2], o que é comum para os nativos digitais. O jogo digital desenvolvido tem o foco na representação virtual de um jogo de voleibol de quadra. O jogo apresenta as seguintes características: força, velocidade e tempo da partida; cálculo da probabilidade de acertos ou erros ocorrerem; a movimentação da bola no plano cartesiano da quadra em que o jogo ocorre; utilizando para movimentação da bola, o cálculo do coeficiente angular da reta ou da tangente do ângulo do triângulo retângulo formado. Logo, permite trazer do mundo real ao virtual o mesmo nível de jogabilidade de uma partida de vôlei.

Assim, utilizando estas abordagens inovadoras unidas ao conteúdo das disciplinas técnicas e propedêuticas do curso, busca-se um maior engajamento dos alunos no desenvolvimento do projeto e respectivos resultados, bem como a sua interdisciplinaridade.

2 MATERIAIS E MÉTODOS

Neste trabalho, o método da pesquisa aplicada será utilizado, pois, um problema específico será abordado e para este uma solução prática proposta e avaliada [3]. Esta abordagem metodológica é alinhada com a formação técnico-tecnológica fornecida pelas escolas técnicas.

Os materiais utilizados no desenvolvimento do jogo foram:

1. JavaScript - A lógica de funcionamento das partidas foram pensadas na linguagem de programação Python, porém devido às questões de implementação da interface visual do jogo foi decidido que o mesmo seria desenvolvido utilizando a linguagem de programação Javascript [5], que permite uma fácil interação com a Web e o uso da API (Interface de Programação de Aplicação) Canvas.
2. API Canvas - A API Canvas [4] permite ao código renderizar/desenhar elementos via Javascript, assim como viabiliza manusear animações com *sprites* de forma simples por meio de linhas de código em JavaScript.

3. POO - A manipulação dos elementos foi projetada a partir do paradigma de programação orientada a objeto (POO) [6]. Cada elemento possui uma classe que quando instanciada é desenhada na tela da API Canvas e sofre alterações conforme imposto pelo código e seus métodos ao decorrer do funcionamento do jogo.

4. Pixel-art - Para que o jogo tivesse uma interface interativa e atrativa para o usuário, foram feitas animações em pixel-art produzidas utilizando o site Pixil-art [8] e pelo software gratuito Paint.net [9]. Trabalhar com pixel-art para jogos é uma forma acessível e prática para criar as animações, pois estas são feitas através de *sprites*, cada *sprite* é uma imagem estática do mesmo objeto em posições diferentes, que juntas formam uma animação como o andar de um personagem, podendo ser inseridas diretamente no código fonte..

5. Matemática - Através da matemática [10] foi possível realizar todas as movimentações dos elementos dispostos na interface, além de gerar todas as possibilidades aleatórias que o programa permite.

3 VOLLEYSRIPT

Para a realização deste trabalho, o estilo de jogo escolhido tem como inspiração os jogos indies dos anos 80, na época, foram criadas as *pixel-arts*, estilo de animação que a tecnologia da época permitia, feita pelos programadores dos jogos, pois era possível fazê-la através das linhas de códigos, ponto a ponto, ou pixel à pixel e, por isso o nome. Hoje em dia, vem como um contraponto às animações 3D e não mais é feita através dos códigos do próprio jogo ou *hardwares* especializados, mas sim por sites como o Pixil-art e *softwares* como Paint.net, que foram utilizados para a produção das animações do jogo, as tornando mais acessíveis e práticas para o animador, sendo econômicas na utilização de recursos onde não se é necessário animações tridimensionais, como é o caso deste trabalho, e por este motivo foi o escolhido para fazer as animações do Volleyscript. As animações que utilizam do estilo *pixel-art* são feitas em formato de *sprites*, que é a junção de imagens estáticas em diferentes posições de um mesmo objeto reunidas sob um mesmo arquivo. As animações como a do andar das personagens para frente, para trás e lateralmente foram feitas cada uma com três *sprites*, sendo uma *sprite* base da personagem parada, e duas em diferentes posições trocando a postura das pernas e dos braços, para dar sensação de movimento. Já as animações das jogadas foram feitas com apenas duas *sprites*, uma para a posição inicial das jogadas e outra para a posição final das jogadas, como saque, recepção e levantamento, diferenciando principalmente a postura dos braços para

simular movimentos como corte, levantamento e saque. A quantidade escolhida de *sprites* foi definida com base na melhor adaptação ao código, sem sobrecarregá-lo, sendo o suficiente para ter uma animação fluida e compreensível para o usuário. A Figura 1 ilustra o jogo desenvolvido utilizando as ferramentas citadas e conceitos.

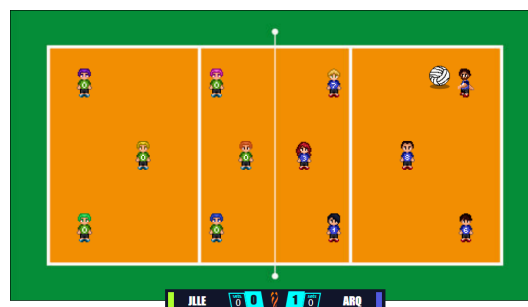


Figura 1: Imagem do jogo desenvolvido.

As *sprites* das personagens são anexadas na ordem desejada que ocorra a animação e colocadas sob um mesmo arquivo, entretanto, cada personagem possui um arquivo contendo suas próprias *sprites*. A Figura 2 ilustra dois grupos de *sprites* para duas personagens diferentes.



Figura 2: Imagem das *sprites* de dois modelos de personagens utilizados no jogo.

O jogo foi desenvolvido aplicando os conceitos estudados na disciplina de POO, em que cada elemento é tratado como um objeto, que possui propriedades (atributos) como posições e métodos que representam as jogadas, o que permitiu trabalhar com pixel-art. As artes foram postas no formato de *sprite*, as *sprites* são um único arquivo contendo todas as artes das movimentações do npc (personagem não jogável - *non-playable character*) facilitando seu uso dentro do código. Através do uso da POO cada *npc* possui propriedades, a sua imagem é uma delas. Para não ser necessário realizar diversos *uploads* de imagens dentro do

código, para cada *npc* é utilizado apenas o arquivo de sua *sprite* e o que mudam são as posições em que o *sprite* está sendo apresentado conforme o que se espera realizar, por exemplo, andar ou receber a bola.

A lógica de movimentação da bola e dos jogadores (objetos), foi criada a partir do que chamamos de “requisições”, ou como implementado no jogo, as “reqs”, que são as posições para onde um objeto é encarregado de ir conforme seu método. Cada objeto possui uma posição e uma requisição. O jogo é formado como um *loop* infinito, em que cada objeto possui um método verificador de posição, que busca a posição do objeto a cada instante e, além disso, há um método para atualizar as posições. O método de verificação examina se a posição do objeto possui os mesmos valores de sua requisição, se não, o método de atualização é acionado. Caso o objeto seja um jogador, a movimentação ocorre de forma simples e direta, em que são adicionados ou subtraídos valores, previamente definidos no código de acordo com a posição da requisição, fazendo com que sua movimentação ocorra de forma linear em apenas um eixo por vez, desse modo as animações são melhor aproveitadas. Com relação ao objeto bola, é esperada uma movimentação oblíqua, nos dois eixos, X e Y, ao mesmo tempo. Para tal, foi utilizado o cálculo da tangente do ângulo do triângulo retângulo que é gerado na relação posição-requisição e, dessa forma, mover a bola na hipotenusa do triângulo [10]. É definido um valor fixo para a alteração do X, que influenciará na alteração do Y. A diferença entre o X inicial da posição e o X da requisição nos informa um cateto, enquanto que a diferença entre o Y inicial da posição e o Y da requisição nos informa o outro. Dividindo o valor da medida do cateto Y pelo valor da medida do cateto X, obtemos a tangente do ângulo do referido triângulo que, multiplicada pelos valores do X que está em movimento com o valor fixo, é aplicada na posição do Y. Assim, o X se move e, conforme sua movimentação, o Y também se altera, gerando o esperado movimento oblíquo sobre a hipotenusa. A Figura 3 exemplifica de forma simples a movimentação explicada.

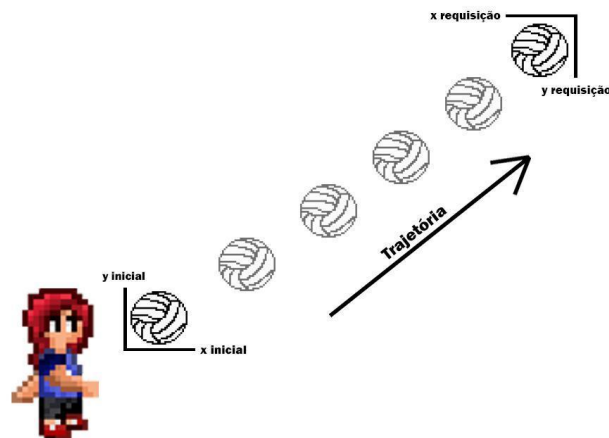


Figura 3: Explicação da movimentação da bola.

Toda vez que a bola para, um evento é acionado, iniciando assim todo o método criado para a lógica do jogo. Na lógica, são verificadas todas as possibilidades como: qual time foi o último a marcar; qual time sacou a bola naquele turno; qual time foi o último a tocar na bola; se a bola caiu no chão; em qual lado da quadra a bola caiu; se a bola sobre a área de um *npc*; o número de toques do time na bola. Dependendo das respostas para as possibilidades, o jogo irá continuar ou parar. Por exemplo, se a resposta para a possibilidade da bola ter caído no chão, for verdadeira (*true*), o jogo finaliza e são realizadas as outras análises; se a resposta para a possibilidade da bola ter parado sobre um jogador for verdadeira, o sistema analisa quantas vezes o time do jogador tocou na bola; se nesse caso for o primeiro toque do time, o jogador realizará o método de recebimento; se for o segundo toque do time, o de levantamento e, se for o terceiro, o de ataque (cortada). Estas jogadas são apenas algumas dentre os diversos cenários gerados pela lógica do jogo que busca o realismo com o jogo de quadra e não digital.

3.1 CLASSES

Para dar suporte a abstração do jogo foram definidas seis classes representando os objetos, sendo elas. Jogo, Bola, Objetos, Posições dos jogadores, Times e Jogador.

1. Objeto: possui o método `RenderSelf()` que renderiza o objeto em questão na API Canvas. Essa classe foi especificada com o intuito de ser herdada por todas as outras que precisem utilizar do método `RenderSelf()` e serem desenhadas na interface, evitando assim, a repetição de código.
2. Bola: herda da classe objeto pois precisa do método para ser renderizada na tela. Possui os métodos para atualizar a

posição da bola e verificar a posição da bola. Porta também a propriedade bola andando, que quando é dada como falsa, o método de atualização da bola aciona toda a lógica do jogo. A Figura 4 ilustra o código em JavaScript da classe Bola, seus atributos e métodos.

```

class ObjetoBola extends Objeto{
  req = {}
  bolaAndando = false
  primeiroMovimento = 0
  XIgualAZero = false
  YIgualAZero = false
  posicaoInicial = {}
  constructor(imagem,x,y,w,h){
    super(imagem,x,y,w,h)
  }
  verificaBolaAndando(){
    if(this.x == this.req.x && this.y == this.req.y){
      this.bolaAndando = false
      this.primeiroMovimento = 0
      jogo.logica()
    }else{
      this.bolaAndando = true
      jogo.limitadorDaLogica = 0
    }
  }
}

```

Figura 4: Código da classe bola: atributos e métodos.

3. Posição: recebe as propriedades X e Y das posições onde o jogador deverá ficar; é atribuída ao jogador.

4. Time: tem como propriedades: seus jogadores, pontos, sets e número da jogada. Seus métodos são basicamente para mudar os valores das propriedades e alterá-los no placar e também realizar o rodízio de seus jogadores.

5. Jogador: recebe uma instância de Posição como propriedade e todas suas possíveis movimentações dentro e seu *sprite*. Possui métodos para saque, recepção, levantamento, corte, realizar as animações com os *sprites* e se movimentar no rodízio.

4.6 Jogo: sua finalidade é fazer o controle de todo o jogo e seus objetos. Seus atributos são o número da jogada do jogo (toques na bola; 1, 2 ou 3), o último time a marcar ponto, o último time a sacar, o último time a tocar na bola, o time que ganhou o último set e o time vencedor do jogo. A partir destes atributos o método da lógica do jogo permite que tudo aconteça, movimentando os elementos e verificando todas as propriedades e possibilidades, além de possuir também métodos para começar um novo set, mudar os lados dos times e do placar, verificar a quantidade de pontos/sets dos times a cada rodada e, por fim, começar uma nova partida.

3.2 MELHORIAS

Para a próxima versão do jogo algumas melhorias devem ser implementadas. Estas melhorias foram apontadas durante a utilização por colegas de turma e respectivas sugestões. Sendo elas:

1. Incluir a força dos jogadores e a velocidade da bola, para utilização no cálculo da probabilidade de acertos.

2. Melhoria do seu *design*. Para a arte final do jogo, serão criados diferentes cenários que irão se alterar entre si ao início de cada nova partida, com temáticas variadas e atrativas ao usuário, e dois novos conjuntos de *sprites*, um para animar o andar diagonal das personagens e um para animar a jogada de bloqueio, ainda inexistente no jogo por ter visto que não haveria necessidade de implementá-la a partida inicialmente para que esta fosse compreensível como uma partida de vôlei ao usuário.

3. Com relação à lógica de funcionamento do jogo, está sendo analisada a possibilidade de ter ao menos dois npcs, um para cada time, jogáveis para o usuário, transformando o jogo que, até então, era apenas entendido como um jogo não-interativo, na oportunidade de utilização para treinamentos de jogadas, principalmente porque, deve-se implementar uma nova lógica para cálculo das probabilidades, baseadas não só nas possibilidades de acerto ou erro, mas também na força aplicada pelos jogadores durante a jogada, somada com a velocidade que a bola atingiu durante o seu percurso.

4 CONSIDERAÇÕES FINAIS

Este artigo teve como objetivo apresentar os resultados da interdisciplinaridade entre as disciplinas propedêuticas de Educação Física, Matemática e Arte e as técnicas no curso técnico integrado em informática, como programação e lógica de programação. Porém, para que o processo desta integração entre as áreas ocorra de forma mais atrativa e interativa, a aplicação de metodologia ativa foi utilizada, em especial a utilização das técnicas de gamificação.

Assim, o desenvolvimento do jogo transcorreu levando em consideração estas ideias e com o respectivo suporte por parte dos professores envolvidos. Para avaliação do mesmo foram feitos testes de usabilidade e jogabilidade com os colegas de turma e um *feedback* foi coletado, o que levou a necessidade de se pensar em uma nova versão atendendo estas demandas. Este mesmo também será apresentado na Semana de Ensino, Pesquisa e Extensão (SEPE) do instituto e novos *feedbacks* serão coletados.

Por fim, foi possível verificar que a abordagem proposta para o desenvolvimento do jogo digital e a respectiva aplicação da metodologia ativa surtiram efeito, os alunos se envolveram com o trabalho, produzindo um ótimo resultado.

AGRADECIMENTOS

Agradecimentos aos professores que transmitiram seus conhecimentos adiante para que fosse possível aplicá-los neste presente trabalho.

REFERÊNCIAS

XIV Computer on the Beach

30 de Março a 01 de Abril de 2023, Florianópolis, SC, Brasil

- [1] Fernandes Barbosa, E. e Guimarães de Moura, D. 2013. Metodologias ativas de aprendizagem na Educação Profissional e Tecnológica. *Boletim Técnico do Senac*. 39, 2 (ago. 2013), 48-67
DOI:<https://doi.org/10.26849/bts.v39i2.349>.
- [2] Carvalho, Ana Amelia Costa da Conceição Soares de. 2019. Jogos, gamification e realidade aumentada: envolver os estudantes na aprendizagem. In *Tecnologias para o português*. 2019;p. 19-25.
URI:<https://hdl.handle.net/11162/201343>
- [3] Andrade, M. *Metodologia Científica*. São Paulo: Atlas, 2006.
- [4] MDN. JavaScript. *Tecnologia Web para desenvolvedores*. [Consult. 20 Out. 2022]. Disponível na internet:
https://developer.mozilla.org/pt-BR/docs/Web/API/Canvas_API
- [5] MDN. JavaScript. *Tecnologia Web para desenvolvedores*. [Consult. 20 Out. 2022]. Disponível na internet:
<https://developer.mozilla.org/pt-BR/docs/Web/JavaScript>.
- [6] MDN. JavaScript. *Tecnologia Web para desenvolvedores*. [Consult. 20 Out. 2022]. Disponível na internet:
<https://developer.mozilla.org/pt-BR/docs/Glossary/OOP>
- [7] Development Guruji. Resurgence of Pixel Art, Internet Archive WaybackMachine, 2016 [Consult. 25 Set. 2022]. Disponível na internet:
<https://web.archive.org/web/20160320075626/http://www.developmentguruji.com/blog/142/resurgence-of-pixel-art.html>
- [8] PIXILART. An art community for everyone!. 2022. Disponível na internet:
<https://www.pixilart.com>
- [9] PAINT.NET. 2022. Disponível na internet:
<https://www.getpaint.net/index.html>
- [10] IEZZI, Gelson. *Fundamentos de Matemática Elementar: Trigonometria*. Volume 3. São Paulo: Editora Atual, 2006.