

Desenvolvimento de um jogo de plataforma na Unity utilizando uma biblioteca de mecânicas

Development of a platformer game in Unity using a mechanics library

Waldecir Martins¹, Fabrizio Honda¹, Fernanda Pires¹, Marcela Pessoa¹

¹Escola Superior de Tecnologia – Universidade do Estado do Amazonas (EST-UEA)
ThinkTED Lab - Pesquisa, Desenvolvimento e Inovação em tecnologias emergentes

{wdsml.lic22, fpires, mspessoa}@uea.edu.br

{fabrizio.honda}@icomp.ufam.edu.br

Abstract. *Digital game development requires multidisciplinary knowledge and involves the repeated use of common mechanics; consequently, the learning curve for game engines like Unity often leads beginners to devote a large portion of their projects to technical implementation. In this context, this tutorial aims to introduce participants to reusable game development in Unity by building a 2D platformer using an open-source library of ready-made mechanics. The adopted methodology is guided: instructors build the game step-by-step while participants replicate each stage, following a process of progressively integrating mechanics. By the end, each participant is expected to have a functional game and be able to apply component reuse in their own projects.*

Keywords *Game development, Unity, Component libraries, Digital games.*

Resumo. *O desenvolvimento de jogos digitais exige conhecimentos multidisciplinares e envolve a implementação recorrente de mecânicas comuns, de forma que a curva de aprendizagem de game engines como a Unity, leva iniciantes a dedicarem grande parte de seus projetos a implementação técnica. Nesse contexto, este tutorial tem como objetivo introduzir participantes ao desenvolvimento de jogos por reuso na Unity, por meio da construção de um jogo de plataforma 2D com uma biblioteca open source de mecânicas prontas. A metodologia adota a codificação guiada, na qual os ministrantes constroem o jogo passo a passo enquanto os participantes replicam cada etapa, em um percurso de integração progressiva de mecânicas. É esperado que, ao final, cada participante tenha um jogo funcional e seja capaz de aplicar o reuso de componentes em seus próprios projetos.*

Palavras-Chave *Desenvolvimento de jogos, Unity, Bibliotecas de componentes, Jogos digitais.*

1. Introdução

O desenvolvimento de jogos digitais não é considerado trivial, pois envolve a articulação de conhecimentos multidisciplinares, como programação, game design, arte e narrativa [Battistella et al. 2014]. Nesse processo, a escolha da game engine tem influência direta na implementação do jogo [Hussain et al. 2020]. Por exemplo, a Unity, uma das game engines mais utilizadas por desenvolvedores [SensorTower 2025], se destaca por sua

versatilidade e suporte. No entanto, pesquisas destacam que seu uso não é simples e pode resultar em uma alta curva de aprendizagem para iniciantes, em razão de sua quantidade elevada de opções, funcionalidades e configurações [Comber et al. 2019]. Somado a isso, podem ocorrer retrabalhos em relação à reimplementação de mecânicas recorrentes e a erros de implementação, assim como desafios na manutenibilidade do código, que pode ser negligenciada em contextos que envolvem prazos curtos [Politowski et al. 2021, Estevam et al. 2023], ressaltando a importância e a dificuldade em implementar jogos.

Nesse sentido, o reuso de *software* é uma estratégia consolidada da Engenharia de Software para resolver esses desafios, permitindo que funcionalidades comuns sejam implementadas uma única vez e aproveitadas em diferentes projetos [Kanode e Haddad 2009]. No contexto de jogos esse reuso pode ser feito através de bibliotecas de componentes, que centralizam mecânicas prontas e configuráveis, reduzindo o tempo de implementação e permitindo que o desenvolvedor concentre esforços no design do jogo [Gazis e Katsiri 2023]. Entretanto, embora existam *assets* e *templates* disponíveis em lojas oficiais, como a Unity Asset Store, parte significativa desses recursos é paga, dispersa e documentada apenas em língua inglesa, o que representa uma barreira adicional para desenvolvedores brasileiros [Ghosh 2022].

Diante disso, considerando a complexidade em implementar jogos e os desafios enfrentados por desenvolvedores iniciantes, este tutorial propõe uma abordagem prática de desenvolvimento de jogos por reuso na Unity, utilizando uma biblioteca *open source* de mecânicas reutilizáveis que centraliza componentes de diferentes gêneros de jogos, com parâmetros configuráveis. Seguindo a organização baseada em componentes da própria Unity, cada mecânica é encapsulada em um componente independente onde o comportamento é ajustado por parâmetros sem alteração de código. Para isso, os participantes constroem um jogo de plataforma 2D, organizado como um percurso de integração progressiva de mecânicas, no qual cada etapa introduz novos componentes da biblioteca e se encerra com um marco jogável.

2. Estrutura detalhada do Conteúdo

Este tutorial tem como objetivo introduzir participantes ao desenvolvimento de jogos digitais na *engine* Unity a partir da reutilização de mecânicas prontas por meio de uma biblioteca *Open Source*. Para isso, ao longo do tutorial será construído, do início ao fim, um jogo de plataforma 2D de modo que os seus elementos, como: (i) movimentação; (ii) coletáveis; (iii) sistema de vida; (iv) obstáculos; (v) plataformas; (vi) inimigos; e (vii) recompensas, sejam providos pelos componentes e *prefabs* disponibilizados pela biblioteca. A Tabela 1 ilustra os tópicos abordados durante o tutorial, bem como sua descrição e estimativa de duração.

Tabela 1. Tópicos a serem abordados no tutorial.

Etapas	Tópico	Descrição	Duração
1	Introdução e configuração do ambiente	Apresentação do contexto de reuso e da biblioteca, criação do projeto Unity 2D e instalação da biblioteca via <i>Unity Package Manager</i> .	40 min
2	Fundamentos do jogo	Construção do personagem jogável com movimentação e pulo, e inserção dos coletáveis com contabilização na interface.	45 min
3	Perigos e sistema de vida	Adição de obstáculos fixos e móveis e configuração do sistema de vida com seus efeitos e interface.	35 min
4	Travessia avançada	Implementação de plataformas móveis e temporizadas e habilitação do salto duplo como progressão do personagem.	35 min
5	Inimigos e combate	Inserção dos inimigos de patrulha e atirador, e configuração dos ataques corpo a corpo e de projétil do jogador.	45 min
6	Finalização do percurso e encerramento	Consolidação do percurso com o baú de recompensa, execução do <i>play test</i> do jogo e recapitulação do processo de construção.	40 min

2.1. Metodologia

O tutorial adota a dinâmica na qual os ministrantes constroem o jogo passo a passo enquanto os participantes replicam cada etapa em suas máquinas. O jogo consiste na construção de um percurso único de plataforma 2D, criado como uma jornada de integração de mecânicas de plataforma, cada trecho do percurso apresenta um conjunto de mecânicas da biblioteca se apoiando nas mecânicas anteriores ao passo que o percurso avança. Dessa forma, cada etapa do tutorial corresponde a um trecho do percurso e se encerra com um marco jogável, onde, ao final de cada bloco os participantes podem executar e testar o caminho construído até então. O percurso é neutro em narrativa, por ser composto por mecânicas genéricas de plataforma, pode ser posteriormente tematizado pelos participantes em diferentes histórias e ambientações, apenas pela substituição dos elementos visuais, sem alteração dos componentes utilizados. Uma imagem das mecânicas reutilizáveis apresentadas no percurso do tutorial, é ilustrado na Figura 1. De modo mais detalhado, em cada etapa será realizada:

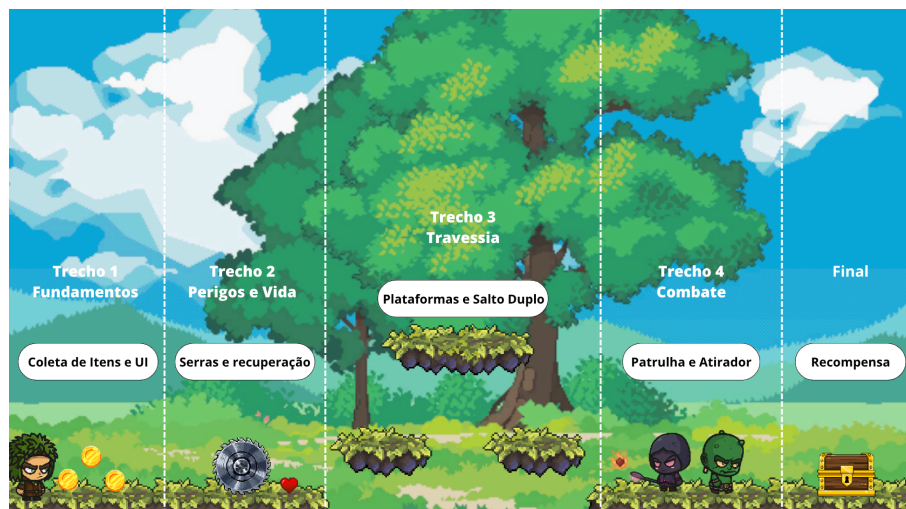


Figura 1. Mecânicas Reutilizáveis por Trecho

- Introdução e configuração do ambiente:** apresentação dos ministrantes, seguida de uma contextualização sobre o cenário de desenvolvimento de jogos digitais, abordando: (i) o crescimento do mercado de jogos e a diversidade de perfis de desenvolvedores; (ii) as game engines mais utilizadas atualmente, como Unity, Unreal Engine e Gogot; (iii) unity como uma das ferramentas mais utilizadas, ao mesmo tempo em que apresenta barreiras técnicas como a curva de aprendizagem, que demanda tempo até mesmo de desenvolvedores experientes; e (iv) o papel da reutilização de *software* nesse contexto, com destaque para o custo da implementação feita do zero e para bibliotecas de componentes como alternativa. A partir dessa discussão, é apresentada aos participantes a biblioteca de mecânicas utilizada no tutorial, em sua organização geral composta por pacotes para os gêneros *platformer*, *topdown*, *tower defense* e *point and click*, e também é apresentado os trechos que serão programados pelos participantes através do uso da biblioteca reutilizável de mecânicas. Após as apresentações, os participantes criam um novo projeto 2D na Unity, instalam a biblioteca por meio do *Unity Package Manager* e realizam uma exploração guiada de sua estrutura, compreendendo o fluxo de trabalho que será repetido em todos os trechos do percurso: (i) inserir o *prefab* ou componente; (ii) configurar seus parâmetros no *Inspector*; e (iii) testar em execução.
- Fundamentos do jogo:** construção do trecho inicial do percurso, no qual o jogador aprende a interagir com o jogo em um ambiente ainda sem riscos. O personagem jogável é montado com os componentes de movimentação lateral e de pulo da biblioteca, ambos com parâmetros configuráveis de velocidade e força, e o cenário inicial é composto. Para dar propósito à exploração, moedas são inseridas ao longo do caminho por meio do componente de itens coletáveis, cuja contabilização é exibida na interface pelo gerenciador do jogo, criando nessa etapa o primeiro ciclo de objetivo e recompensa. Ao final do trecho, o jogador domina o deslocamento e a coleta, e o percurso está pronto para receber seus primeiros desafios.
- Perigos e sistema de vida:** construção do trecho onde após a exploração aprendida passa a ter consequências. São adicionados obstáculos com o

componente de serras, capaz de operar em configuração fixa ou móvel, exigindo que o jogador aplique a movimentação dominada no trecho anterior para desviar deles. Como agora é possível falhar, o sistema de vida do personagem é configurado com o componente responsável por dano, cura e morte, acompanhado da interface de corações e dos efeitos visuais correspondentes. Fechando o ciclo, o coletável de vida é inserido pelo caminho, reaproveitando a mecânica de coleta do trecho anterior, agora com a função de recuperação.

- **Travessia avançada:** construção do trecho que necessita de experiência com a etapa de movimentação, nessa etapa jogador atravessa áreas de queda que exigem a utilização dos componentes de plataforma, sendo eles: (i) plataforma móvel, que transita entre pontos; (ii) plataforma móvel ativada por tecla; e (iii) plataforma temporizada, que faz a superfície desaparecer e reaparecer após o contato. Como recompensa pela travessia, o salto duplo é habilitado por meio do parâmetro correspondente do componente de pulo ampliando a capacidade de movimentação do personagem, que será exigida no trecho seguinte e ilustrando para os participantes a possibilidade de configurar facilmente os componentes.
- **Inimigos e combate:** construção do trecho em que os perigos reagem ao jogador. O primeiro inimigo é montado com o componente de patrulhamento entre pontos e o componente de dano por contato, sendo enfrentado de perto com o ataque corpo a corpo do jogador. O segundo inimigo utiliza o componente de inteligência de tiro, com raio de detecção e cadência de disparo configuráveis, forçando o enfrentamento à distância com o componente de disparo de projéteis do jogador. Ambos os inimigos recebem o mesmo sistema de vida aplicado ao personagem nos trechos anteriores, agora com suas respectivas barras de vida, evidenciando para os participantes como um mesmo componente da biblioteca pode ser reutilizado em diferentes papéis.
- **Finalização do percurso e encerramento:** consolidação do trecho final em que a jornada se completa com o jogador alcançando a recompensa de conclusão, implementada com o componente de baú de recompensa da biblioteca. Os participantes realizam ajustes finais de *level design*, como posicionamento de coletáveis, obstáculos e pontos de dificuldade, e executam o *play test* do jogo, percorrendo do início ao fim o percurso e os desafios construídos ao longo da sessão. Para encerrar, é feita a recapitulação do processo, evidenciando as mecânicas obtidas por reuso e o esforço de implementação evitado, seguida da sessão de dúvidas e discussão.

3. Discussão e Contribuições Esperadas

O tutorial busca apresentar de forma prática, o desenvolvimento de jogos através de uma biblioteca de mecânicas reutilizáveis, no qual os participantes vivenciam a construção de um jogo a partir de componentes prontos e configuráveis. A organização em um percurso que integra novas mecânicas continuamente, com marcos jogáveis ao final de cada etapa, permite que participantes de diferentes níveis de experiência acompanhem a construção e verifiquem seu progresso.

Como contribuições espera-se que os participantes concluam o tutorial com um jogo de plataforma 2D funcional e sejam capazes de aplicar o fluxo de reutilização de componentes em seus próprios projetos, em diferentes temáticas e contextos. Para

a comunidade de jogos, o tutorial contribui ao apresentar uma ferramenta gratuita, de código aberto e com documentação em língua portuguesa, que pode ser adotada em disciplinas, oficinas e projetos de pesquisa, ampliando o acesso de desenvolvedores iniciantes a recursos de implementação. Por fim, a realização da atividade permite coletar percepções dos participantes, que poderão orientar a evolução da biblioteca e de suas futuras versões.

Referências

- Battistella, P., von Wangenheim, C., e Fernandes, J. M. (2014). Como jogos educacionais são desenvolvidos? uma revisão sistemática da literatura. In *Workshop sobre Educação em Computação (WEI)*, pages 159–168. SBC.
- Comber, O., Motschnig, R., Mayer, H., e Haselberger, D. (2019). Engaging students in computer science education through game development with unity. In *2019 IEEE Global Engineering Education Conference (Educon)*, pages 199–205. IEEE.
- Estevam, V. B., dos Santos, A. D., e Paixao, M. (2023). Design patterns and code maintainability in games: A case study. In *Simpósio Brasileiro de Jogos e Entretenimento Digital (SBGames)*, pages 299–304. SBC.
- Gaziz, A. e Katsiri, E. (2023). Serious games in digital gaming: A comprehensive review of applications, game engines and advancements. *arXiv preprint arXiv:2311.03384*.
- Ghosh, A. (2022). Challenges in educational game development. *International Journal of Darshan Institute on Engineering Research & Emerging Technology*, 11(1):54–60.
- Hussain, A., Shakeel, H., Hussain, F., Uddin, N., e Ghouri, T. L. (2020). Unity game development engine: A technical survey. *Univ. Sindh J. Inf. Commun. Technol.*, 4(2):73–81.
- Kanode, C. M. e Haddad, H. M. (2009). Software engineering challenges in game development. In *2009 Sixth International Conference on Information Technology: New Generations*, pages 260–265. IEEE.
- Politowski, C., Petrillo, F., Ullmann, G. C., e Guéhéneuc, Y.-G. (2021). Game industry problems: An extensive analysis of the gray literature. *Information and software technology*, 134:106538.
- SensorTower (2025). The big game engine report of 2025. Technical report, SensorTower.