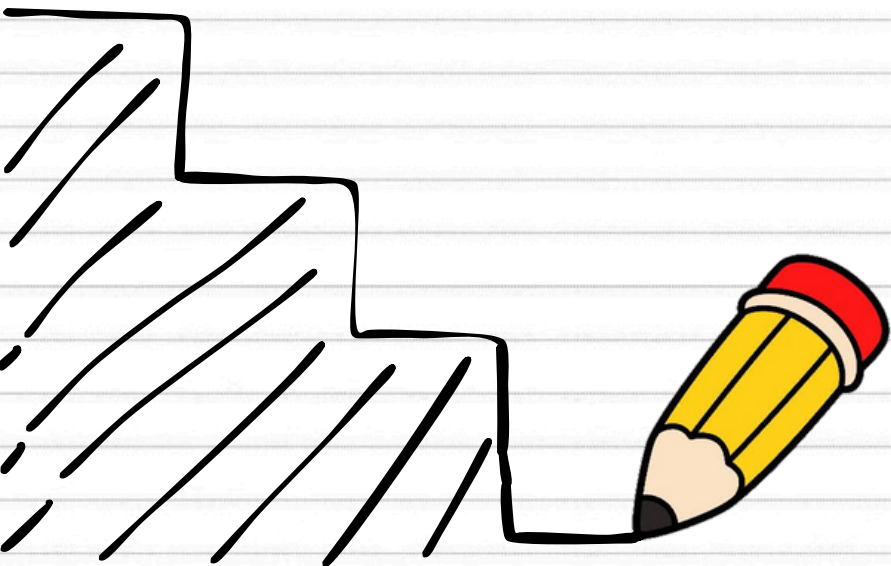


# GAME DESIGN

O processo de criação de um jogo

Itallo Casimiro





# Sinopse

Conheça os fundamentos da área de design de jogos e de seus primeiros passos na idealização de um jogo com o ebook "Game Design, o processo de criação de um jogo". Este ebook foi escrito com o objetivo de ensinar de forma clara, sem termos técnicos, para que o leitor consiga entender sem problemas a como dar seus primeiros passos na área de desenvolvimento de jogos

Este ebook é dividido em cinco módulos, indo do conceito de design de jogos até o começo da idealização de um jogo, além de abordar brevemente sobre as tecnologias de programação, com imagens representativas para a melhor compreensão, é um ebook curto, apenas uma abordagem inicial, existem diversos outros conceitos que podem ser estudados da área principalmente na área da programação

# Sobre o autor

Itallo Casimiro é um garoto estudante da área de programação de desenvolvimento de jogos, tanto da área de game design como de programação, planejando desenvolver um jogo inovador no futuro com uma equipe sólida de desenvolvedores, trabalhando como game designer da equipe



@itallo\_casimiro



Itallo Casimiro Felix



Itallo Casimiro



@itallo\_casimiro

# módulo 1

## Introdução ao game design

Seja introduzido a área do design de jogos e entenda o que faz o profissional

# O que é o Game Design ?

O game design é uma área que faz parte do desenvolvimento de jogos, o profissional, o game designer, ao contrário do que muitos pensam, não é responsável pela arte do jogo, mas sim pela criação da essência do jogo, isto é, a história do jogo, personagens, mecânicas, inimigos, itens, entre outros fatores que compõem o jogo. O game designer também é responsável por criar a documentação do jogo e se comunicar com toda a equipe de desenvolvimento, de forma clara e objetiva, para garantir uma boa compreensão para cada membro da equipe

## Áreas do design de jogos

- Game Director
- Game Designer
- Technical Game Designer
- Level Designer
- User Interface Designer (UI Designer)
- User Experience Designer (UX Designer)
- Sound Designer

## Game Director

São chamados para auxiliar na criação de design de personagens, na criação de novas animações e testes de unidade, são chamados para codificar a engine básica de acordo com as ideias da equipe de design

# Technical Game Designer

Está ligado a criação das interações, dinâmicas, ações e outros pontos que podem ser encontrados no UX Designer, duas áreas que geralmente trabalham em conjunto nesse quesito

# Level Designer

Responsável pela criação dos níveis de jogo, o ambiente onde o jogo ocorrerá, localização dos inimigos e itens, com objetivo de trazer imersão na jogabilidade do jogo para torná-lo desafiador

# UI Designer

Responsável pela criação da interface onde o usuário poderá interagir, como telas de menu, start, game over, entre outros

# UX Designer

Cria a interação entre o usuário no jogo, é responsável por perceber a forma como o jogador vai se comportar para oferecer a melhor experiência possível dentro do jogo

# Sound Designer

Responsável pela criação da trilha sonora do jogo, a música, sons, barulhos, ruídos, melhorando ainda mais o UX do jogo e adicionando mais detalhes que, conseqüentemente, prendem mais o jogador

# Habilidades, e responsabilidades

No desenvolvimento de jogos, o Game Designer possui uma série de fatores como profissional e conhecimentos para se especializar na área. Entre as habilidades temos a criatividade, pois não adianta apenas sentar em uma cadeira e pensar por horas e horas em uma boa ideia, você também precisa de uma criatividade, uma fonte de ideias, despertar a criatividade que está em você, pois sem ela, possivelmente nada do que você planejar pode dar certo por inúmeros motivos, como falta de relação entre as ideias, e de nada adianta ter uma boa ideia se essa ideia não se encaixa no planejamento inicial do jogo

Além da criatividade, um Game Designer precisa ter conhecimentos breves sobre cada área do desenvolvimento de um jogo, na parte da programação, som, arte, e todas as áreas que envolverão a criação do seu jogo, principalmente conhecimento sobre a equipe em que você vai trabalhar durante o desenvolvimento. Isso não quer dizer que o Game Designer deve saber tudo sobre cada área, mas saber o básico sobre as áreas da sua equipe é uma boa maneira de conseguir pensar a longo prazo, de forma com que de certo no final

- Criatividade
- Conhecimento em programação
- Resolução de problemas
- Gerenciamento de tempo
- Atenção a detalhes
- Comunicação



Um Game Designer é o tipo de profissional que está sempre aberto a ouvir novas ideias por parte de sua equipe de desenvolvimento, testando e validando ideias, essas ideias então são colocadas em um documento que possui as informações do jogo. O Game Designer é o que chamamos também de "ponte de comunicação", é ele quem se comunica com a equipe para falar sobre as ideias propostas tanto por ele como por outros membros

É de extrema importância que o Game Designer conheça o mercado de jogos, para poder produzir o melhor jogo possível aos jogadores que irão comprar seu jogo, e também para poder analisar bem as tendências de jogos, obter conhecimentos de público-alvo, e também para poder conseguir referências para o seu jogo, o que é outro ponto que deve ser tratado com cuidado

## Referências

Só começar a desenvolver um jogo, uma boa forma para ter ideias é ter uma fonte de referências, principalmente em outros jogos, isto é uma boa forma de conseguir chegar ao resultado esperado sem ter que "se esforçar do 0", porém é necessário ter os devidos cuidados ao buscar por referências e usá-las, evite ao máximo copiar as características marcantes da referência que você buscou, caso contrário, isso pode acabar movendo a comunidade a acusar o jogo de plágio levando o jogo a ter uma fama cada vez mais escassa. As referências são apenas uma base para você ter uma ideia, como se fosse um esqueleto, e perceba que todo e qualquer jogo tem referências, sejam elas perceptíveis ou não. Lembre-se que as referências não são apenas sobre arte e desenho, elas também são usadas para desenvolver mecânicas novas, sistemas e até mesmo a história do jogo





# O que são frameworks ?

No Game Design, os frameworks são ferramentas que ajudam na orientação do processo criativo de um jogo, ajudando a guiar no processo de desenvolvimento de um jogo, acompanhando no desenvolvimento até o lançamento. Como exemplo de framework temos a MDA ( Mecânicas, Dinâmicas e Estéticas)

Conhecer e estudar os frameworks trazem diversos benefícios durante o desenvolvimento, ajudando a gerenciar complexidade, evitar erros comuns e até mesmo economizar tempo, garantindo uma boa qualidade no lançamento do seu jogo para que atenda às expectativas dos jogadores. Além de tudo, os frameworks oferecem uma melhor comunicação, facilitando a colaboração entre os desenvolvedores, garantindo trabalhar seguindo em uma mesma direção

## Jogos INDIE e AAA

Os jogos INDIE, ou jogos Independentes, são jogos feitos geralmente com um orçamento pequeno, por uma equipe pequena, ou até mesmo por uma pessoa só, como Braid, Fez e Super Meat Boy

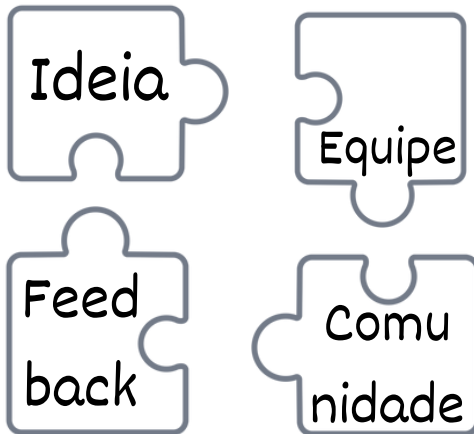
Os jogos AAA (Triple A) são jogos desenvolvidos por uma equipe grande e com um grande orçamento que demoram anos para serem produzidos, como God Of War, GTA e Mortal Kombat

## Existe ideias ruins para jogos?

Durante o desenvolvimento e até antes podemos ouvir de outras pessoas que a ideia é ruim ou que não vai dar certo, porém na prática não é bem assim, não existem ideias ruins para jogos, mas sim ideias más executadas dentro do jogo. Uma ideia, de repente, que parece ruim pode ser capaz de criar até mesmo um novo gênero de jogo, então confie na sua ideia e implemente da maneira certa

# Como validar ideias ?

Validar uma ideia para o seu jogo é algo crucial para evitar o desperdício de tempo e até mesmo dinheiro. O primeiro passo, claro, é fazer uma pesquisa de mercado, buscar por referências em um mercado de jogos como a steam pode desencadear incontáveis ideias, possibilitando identificar as novas e atuais tendências do mercado para ter o melhor resultado possível no lançamento do seu jogo e em seguida, basta apenas anotar suas ideias e passa-las inicialmente para um papel e idealizar a mecânica principal. Após esse processo, busque por aprovação da sua equipe e da comunidade disponibilizando um conteúdo referente ao seu jogo, para que eles possam avaliar a jogabilidade e os conceitos iniciais do jogo



## O que é High Concept ?

Este termo é muito usado na área de Game Design, o High Concept é uma documentação, usada em jogos de escopo médio e grande. Geralmente, é uma documentação mais objetiva para validar ideias, se trata de um resumo do jogo e seus principais pontos que serão apresentados aos jogadores

Essa documentação deve ser escrita da melhor maneira possível, pois é ela que vai apresentar as ideias a comunidade de jogadores. Um detalhe importante é que ela deve ser escrita em uma linguagem com o menor número de termos técnicos possíveis, isto é, escrito de forma com que todo e qualquer leitor consiga entender sem dificuldades e que consiga compreender exatamente a proposta do jogo

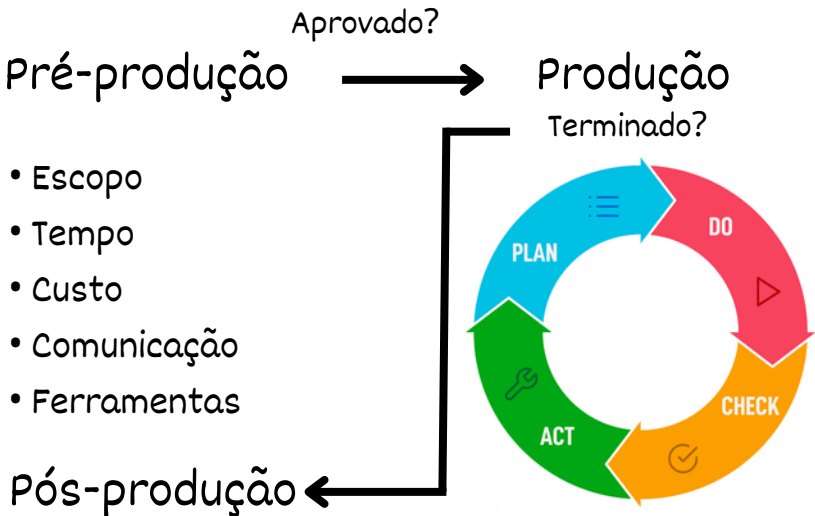
O High Concept é feito antes do início do desenvolvimento de um jogo e antes mesmo da documentação final do jogo, o Game Design Document, sendo uma documentação curta, entre duas páginas

## Montando High Concept

1. Crie uma pequena descrição atrativa do seu jogo
2. Faça uma lista e explique brevemente sobre as mecânicas principais do seu jogo
3. Fale sobre a história do jogo, quem são os personagens, ambiente onde se passa o jogo, entre outros
4. Coloque informações sobre onde o jogo será publicado, as plataformas onde será lançado
5. Fale sobre a equipe de desenvolvimento de modo breve mas claro
6. Cite a engine de desenvolvimento do seu jogo

# Organizando as ideias

No começo do processo criativo, podemos ter ondas imensas de ideias, porém essas ideias em excesso podem fazer-nos perder o controle, por isso o ideal é dividi-las em blocos PDCA, Plan, Do, Check, Action ou, em português, Planejar, Fazer, Checar, Agir. Esse processo ajuda principalmente a não nos perdermos nas nossas próprias ideias, mas não quer dizer que você não possa pensar fora desse esquema, é recomendado que você anote todas as suas ideias e as execute no momento certo



## Errado:

Planejar o jogo inteiro, todas as mecânicas, música e sons, história... E implementar tudo de uma vez

## Certo:

Dividir em blocos, por exemplo:

1. Defina as mecânicas básicas, uma fase e implemente-as no jogo, teste-as e valide-as
  2. Comece a desenvolver a narrativa
- [...]

# Game Engines

As game engines são os softwares usados no desenvolvimento de jogos que fornecem diversos recursos que facilitam a criação de um jogo de forma rápida e eficiente, como a fácil importação de objetos 2D e 3D, adicionar luzes, física, áudios, entre outros recursos de forma simples e prática

Existem diversos motivos que explicam o motivo dessas Engines serem tão utilizadas no mercado de jogos, como a agilização de trabalho visual, que permite a importação, renderização e configuração de todo e qualquer asset do seu jogo, os áudios que são importados em alta qualidade em diversos formatos, entre outras muitas funcionalidades

As Engines são responsáveis por dar vida ao seu jogo, através da programação, o algoritmo será responsável por criar as funções dentro do seu jogo, todas as interações. Grande parte das Engines usam o sistema de scripting para isso, porém algumas usam de forma visual e lógico, sem códigos, como o Construct 3, que tem suporte a programação JavaScript, no entanto o padrão é o scripting visual, com blocos de comando

1	System	Every tick	detector	Set position to ( <i>monster.ImagePointX("detector")</i> , <i>monster.ImagePointY("detector")</i> )
Add action				
2	monster	Is animation "Right" playing	monster	Set animation to "Left" (play from beginning)
	detector	Is overlapping  tile	monster	Simulate  Platform pressing Left
Add action				
3	monster	Is animation "Left" playing	monster	Set animation to "Right" (play from beginning)
	detector	Is overlapping  tile	monster	Simulate  Platform pressing Right
Add action				
Add event				

Entretanto, nos dias de hoje, as Engines mais usadas no desenvolvimento de jogos são: Unity, Unreal, Game Maker e a Godot, que já usam um sistema de scripting em formato de código

# Erros comuns no Game Design

Durante a produção do escopo de um jogo podem haver algumas ou várias complicações que acabam fazendo com que o projeto seja abandonado ou seja um fracasso no mercado. Entre esses erros, os principais geralmente são:

1. Fazer um escopo muito grande sem ter experiência na área
2. Não testar o próprio jogo
3. Criar mecânicas muito complicadas
4. Não ouvir feedbacks da comunidade
5. Fazer um jogo mal balanceado

## O que são NPC's ?

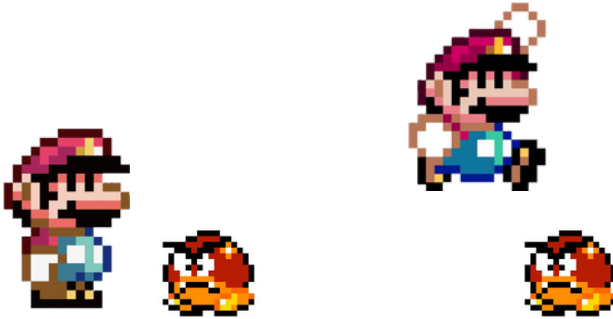
Os NPC's (Non-Playable Character) foram criados para serem figurantes dentro da narrativa do jogo, ou seja, personagens que compõem o cenário, aqueles personagens não jogáveis

## Tutoriais e comandos de jogo

A forma como você ensina o jogador a jogar o seu jogo influencia na decisão dele, se quer continuar jogando ou não. Se você mostrar um texto enorme ensinando um comando de jogo, é quase certo que o jogador perderá o interesse em jogar o seu jogo, então uma boa solução para isso é ensinar os comandos na prática, uma submecânica, isso fará com que o jogador aprenda sozinho como usar esse comando e pensar sobre as situações em que deve ser usado, sem a utilização de um grande texto para isso, o que vai incentivar o jogador a explorar mais essa mecânica durante o jogo

# O que é jogabilidade ?

A jogabilidade pode ser entendida como um conjunto de mecânicas que se comunicam. Como em Super Mario, em que você pode derrotar um inimigo pulando sobre ele, porém se encostar de frente com ele é game over, isso em si pode ser considerado uma jogabilidade, você deve usar a mecânica de pular para derrotar um inimigo



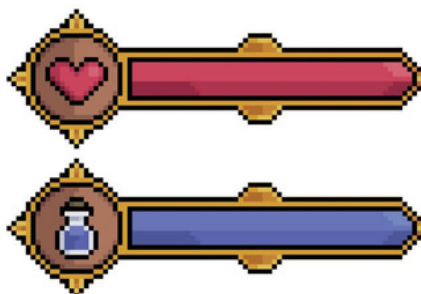
# O que é a fadiga visual ?

Este termo não se refere diretamente a arte de um jogo, mas sim ao tempo em que o jogador passa em um único cenário sem novidades, inimigos e perigos repetitivos, e isso causa a fadiga visual, que podem levar um jogador a desistir do jogo, encontrado até mesmo em grandes jogos. Por isso é importante sempre estar introduzindo novidades até mesmo no ambiente do jogo, seja com histórias, Power ups, ou qualquer elemento novo que chame a atenção do jogador

# O que são Huds ?

São elementos na interface do jogo que apresentam elementos que podem ser facilmente interpretados pelo jogador, como as próprias barras de vida e energia, acompanhadas de uma bela harmonia de cores que não atrapalhe o jogador durante a gameplay mas que

consiga mostrar as informações explicitamente assim que forem reparadas



## Jogos VR e AR

Os jogos VR (Virtual Reality), ou jogos de realidade virtual, são uma experiência que desfruta da simulação real, como se o jogador estivesse dentro do jogo. O VR captura, em geral, movimentos do jogador com câmeras para rastrear marcadores com luzes LED no corpo do jogador

Os jogos AR (Augmented Reality), ou jogos de realidade aumentada, sobrepõem elementos virtuais a nossa realidade, um exemplo disso é o jogo Pokémon GO, que usa a localização GPS para gerar elementos virtuais de acordo com a localização do jogador



## módulo 2

# Game Design Document

Conheça a documentação feita por um game designer, o GDD

# O que é o GDD ?

O Game Design Document ou GDD é um documento que tem como objetivo planejar as ideias de um jogo apresentando a proposta do jogo e o seu conceito. O tamanho do GDD varia de acordo com dois fatores:

- Tamanho do projeto : quanto maior o projeto, maior a quantidade de informações no GDD
- Tamanho da equipe : quanto maior a equipe, maior a quantidade de detalhes no GDD

A qualidade do GDD depende da capacidade do Game Designer de apresentar a proposta do jogo de forma clara e satisfatória, por meio de imagens, textos, desenhos, planilhas, mapas mentais e entre outros métodos. O GDD pode conter:

- Gênero do jogo
- Público-alvo
- plataforma (s)
- DVs (Diferenciais de Venda)
  - "50 armas para colecionar"
- Narrativa (história)
- Mecânicas
- Estilo artístico
- Referências
- Personagens
- Diálogos e Cutscenes
- Interface
- Recursos (logo, som etc...)
- Escopo
- Estética
  - Emoções
  - Atmosfera
  - Tipografia
- Level Designer
- Trilha sonora

# Tipos de GDD

## • GDD de página única

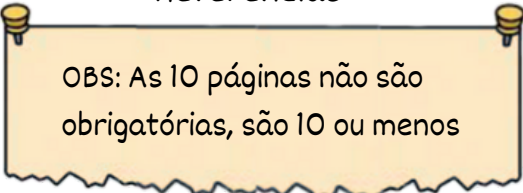
Este Game Design Document apresenta informações básicas do jogo, como:

- Nome do jogo / logo
- Público-alvo
- Gênero
- Resumo
- DVs
- Plataforma(s)
- Referências
- Modos de gameplay

## • GDD de 10 páginas

Este Game Design Document mostra seu jogo e sua completude, mas sem entrar em detalhes, contendo informações como:

- Nome do jogo / logo
- Público-alvo
- Gênero
- Plataforma(s)
- História
- Personagens
- Controles
- Gameplay
- Mundo do jogo
- Escopo
- Mecânicas
- Inimigos
- Diálogos
- Referências



OBS: As 10 páginas não são obrigatórias, são 10 ou menos

## • Gráfico de ritmo

Este Game Design Document mostra o progresso do jogo, como uma tabela, que pode conter informações como:

- Nome do nível
  - Horário
  - Elementos da história
  - Progressão do personagem
  - Tempo de jogo
  - Inimigos
  - Mecânicas
- GDD bíblia

Este Game Design Document contém todas as informações do jogo e todos os detalhes possíveis, o GDD completo

## O que são mecânicas e submecânicas ?

As mecânicas podem ser descritas como "ações", como pular, atacar, luzes, tiros, entre outros. Basicamente, as mecânicas são interações, e eventos que ocorrem durante o jogo, por exemplo, quando você se cura em um jogo, isso é uma mecânica de cura, quando você fala com um personagem, isso é uma mecânica de diálogo, e assim por diante

As submecânicas são a maneira como o seu jogo consegue adaptar a mecânica principal em momentos diferentes do jogo, para trazer constantemente novos desafios para o jogador. Por exemplo, quando você tem uma mecânica de pular em um jogo e encontra um relevo de chão em que você só pode subir pulando, isto é uma submecânica, uma forma de usar a mecânica principal em uma situação no jogo

# O que é escopo de jogo ?

O escopo do jogo é um tópico encontrado no GDD que apresenta as propostas no lançamento de um jogo, promessas de entrega, marketplace, idiomas de tradução, entre outros, tudo o que os desenvolvedores prometerão a comunidade do jogo no lançamento, por exemplo:

- Este projeto visa em criar um jogo [...], seus entregáveis incluemem:
    - Design de personagens e cenários
    - Lançamento na Steam
    - Tradução para outros idiomas
    - Versões para outras plataformas
- [...]

No escopo, também pode ser citado algumas mecânicas do jogo que são julgadas inovadoras, que realmente vão chamar atenção do público-alvo

Portanto, escopo de um jogo é um conceito que define a grandeza e a complexidade de um projeto

# O que é o storytelling ?

O Storytelling é a arte de contar histórias de forma envolvente ao leitor, de forma com que ele se interesse pela história contada, bastante utilizada em campanhas de marketing, publicidade e entretenimento. O Storytelling foca em desencadear emoções no leitor durante o conto, transformando emoções em narrativas cativantes. Criar uma boa história é geralmente um desafio na área de desenvolvimento de jogos, mas existem sim dicas para criar uma boa narrativa envolvente, entre elas, temos:

- Conheça o seu público-alvo
- Defina o objetivo da sua história
  - o que você quer que os jogadores sintam
  - qual ou quais sentimentos você quer passar
- Estructure uma história, com:
  - introdução: apresente os personagens
  - Conflito: crie o problema que deve se resolvido
  - Clímax: ponto onde o problema fica complicado
  - Desfecho: termine a história com um bom final



## a importância do storytelling

A narrativa do seu jogo influencia principalmente na construção e nas características do seu mundo, é o Storytelling que vai combinar os elementos de jogo com a ambientação de um nível. Ele também é responsável pela interconexão dos mapas, trazendo uma maior integração entre os cenários do seu jogo

# O que é Brainstorming ?

O Brainstorming é uma das técnicas para ter ideias, consiste em basicamente se reunir com um grupo de pessoas ou com a sua própria equipe e começar a pensar em um monte de ideias diferentes e fazer anotações de todas elas e discutir sobre cada uma delas com a equipe para aprovar ou rejeitar a ideia, em uma tradução literal, significa "tempestade de ideias". Essas ideias podem envolver qualquer coisa relacionada ao jogo, seja na criação de movimentações, telas de menu, mecânicas, entre outros

módulo 3

# Criando um bom Level Design

Conheça os fundamentos sobre o Level Design e aprenda a criar um nível de jogo



# O que é o Level Design ?

O Level Design é a área responsável pela criação das fases do jogo, o ambiente, interações, desafios, entre outros. O Level Design precisa representar essas criações em elementos de cenário, o que faz seu jogo fazer com que um jogo fique rico em detalhes e imersão

A interação das mecânicas com as fases e mapas proporciona uma experiência de jogo mais divertida, a aplicação das submecânicas. O Level Design propõe ao jogador possibilidades e dificuldades para fazer algo, o que acaba prendendo o jogador na experiência do jogo. Essa sem dúvidas é uma das partes mais cuidadosas do desenvolvimento de um jogo, pois é ela quem vai ter a capacidade de chamar a atenção do jogador na experiência



# O que é o espaço de jogo?

O espaço de jogo é onde ocorre o jogo. O espaço não é necessariamente algo físico, um ambiente físico, em um jogo de perguntas e respostas por exemplo, você não tem um espaço físico, um cenário, existe apenas o jogador que está respondendo a uma pergunta que foi emitida que vai verificar se a resposta está certa ou errada, esse é o espaço dentro desse jogo

# Elementos, atributos e estados

Os elementos são tudo aquilo que temos que lidar dentro do jogo, como personagens, itens, entre outros

Os atributos são as características desse elemento, o nível de um personagem é um atributo por exemplo, assim como energia força destreza, entre outros

Os estados são informações adicionais sobre os atributos que eu tenho em um personagem, como envenenado, paralisado, fraco, estado que o personagem pode assumir durante o jogo

## Dicas para um bom Level Design

Um bom Level Design varia de jogo para jogo, depende do gênero, objetivos, histórias, mas uma coisa não muda de jogo para jogo, você deve achar um jeito de deixar o jogador entretido no seu jogo, aí que o Level Design entra, para isso, temos algumas dicas para estruturar um bom Level Design:

- Adicione caminhos secundários e áreas secretas
- Deixe recompensas pela exploração
- Crie narrativas implícitas através do ambiente
- Permita o jogador explorar
- Faça progressão de novas mecânicas
- Equilibre a dificuldade do jogo
- Não deixe o personagem andar por muito tempo
  - Adicione obstáculos e inimigos
  - Itens
  - Lugares novos para explorar

- Aumente o tamanho do nível de acordo com a progressão
- Trabalhe nas emoções e na atmosfera
- Evite criar paredes invisíveis
- Não coloque o jogador em risco logo no início
- Sempre introduza novidades, em momentos diferentes, como novas mecânicas
- Teste suas fases com várias pessoas diferentes

## Micro x Macro Design

O micro design é o foco em detalhes internos de um elemento, nas fases e mapas de jogo. Ao invés de abordar o design do mapa inteiro, o micro design divide eles em unidades menores, como salas e seções dentro do nível de jogo, permitindo uma análise mais detalhada e cuidadosa, posicionando melhor os inimigos, obstáculos e itens por essa divisão do mapa, criando uma melhor experiência de jogo naquela fase

O macro design, por outro lado, foca em como os diferentes níveis de jogo se conectam no mapa, tratando do fluxo e estrutura global do mapa do jogo. Contribuem principalmente para a experiência do jogador, conectando as salas do micro design, criando uma progressão e distribuição de desafios entre os níveis do jogo, além de criar diferentes sentimentos em cada parte do nível, auxiliando em uma aventura imersiva e emocionante

## Como incentivar a exploração?

Existem muitas formas de fazer com que o jogador sinta vontade de explorar dentro do seu jogo, uma delas é deixar com que o jogador decida a forma que quer explorar, basicamente, permita com que o

jogador veja outra forma de progredir no seu jogo de forma que não seja obrigado a fazer algo, colocando uma recompensa no final, um diálogo, ou qualquer coisa que faça com que o jogador tenha vontade de explorar os locais

## Cenários incríveis para seu jogo

Os cenários, na maioria das vezes, são feitos com imagens estáticas, que estão ali paradas apenas para compor o background, isso de certa forma funciona em muitos jogos, porém se quisermos realmente um cenário detalhado e bonito, que encante o jogador, podemos adicionar elementos únicos, elementos repetitivos, como árvores, podem ser feitas uma diferente da outra para detalhar mais o cenário, além da adição de animações, sons e música que auxiliam ainda mais na composição do cenário

## Feedbacks dentro de um jogo

Neste caso, quando falamos de feedbacks, nos referimos à uma resposta a ação do jogador, que gera um sentimento no jogador. Por exemplo, em um jogo quando batemos em uma parede indestrutível, percebemos que não é possível destruí-la, ou seja, um feedback. Sabendo disso, crie um bom feedback, adicione sons, partículas, tremor na câmera, ou qualquer outro complemento para esse feedback ficar explícito e bacana ao jogador

módulo 4

# Introdução a linguagem C# com a Unity Engine

Seja introduzido brevemente a programação  
com a linguagem c# com a Unity Engine

# O que é a Unity ?

A Unity é um motor de jogo (Engine), proprietário criada pela Unity technologies. A Unity é uma ferramenta que permite criar jogos 2D e 3D para diversas plataformas, como PC, consoles, mobile, VR e AR, utilizando um editor virtual e programação com as linguagens C# (C-Sharp) ou C++

## Projetos 2D URP e 2D HDRP

Os projetos 2D URP são projetos que serão feitos com o intuito de serem jogados em todas as plataformas e proporcionar uma ótima performance e gráficos

Os projetos 2D HDRP são focados na produção de projetos para a última geração da tecnologia, de forma "poderosa", ou seja, jogos pesados, possui uma dificuldade maior para se dominar



## Interface da Unity

Assim que você cria um projeto, será redirecionado a uma interface onde você começará a criar o seu jogo. Nessa interface temos algumas ferramentas, como:

- Scene: Tela onde o jogo é montado
- Hierarchy: Lista de elementos dentro da cena

- **Inspetor:** Permite configurar um elemento de cena, como posição, layers, tamanho, entre outros
- **Project:** Guarda os arquivos do jogo, como assets, packages, entre outros
- **Console:** Alerta sobre erros e códigos de atenção, e serve para testar seu(s) script(s)
- **Game:** Tela onde o jogo vai rodar
- **Game Object:** Cria objetos de jogo, como sprites por exemplo

## Importação de arquivos do seu PC

Para importar imagens, áudios, e outros arquivos, faça isso:

- Abra o gerenciador de arquivos
- Procure o arquivo e arraste até a parte de assets dentro da Unity (pastas de arquivos)
- Ou abra a assets store dentro da Unity na aba window (necessário Unity ID)

# Linguagem de programação C#

O C# (C-Sharp) é uma linguagem desenvolvida pela Microsoft que possui uma tipagem forte, sendo uma linguagem orientada a objetos. O C# permite que as aplicações feitas com a linguagem possam ser executadas em diversos dispositivos. Utilizada principalmente em aplicativos Windows e desenvolvimento de jogos com a Unity

## Variáveis

As variáveis são espaços na memória do computador que armazenam dados que podem ser alterados durante a execução do código. Para declarar uma variável, devemos seguir o seguinte:

**Tipo Nome = Valor ;**

As variáveis podem ou não serem declaradas com um valor inicial, mas lembre-se o valor deve ser do mesmo tipo da variável se não será retornado um erro, por exemplo:

```
int hp = 10;
```

OBS: Qualquer script no C# DEVE terminar com ponto e vírgula (;) no final da declaração!  
Variáveis não podem ser nomeadas com caracteres especiais ou espaços, e números apenas depois de uma letra





# Tipos de dados

- int: Dados de tipo numérico inteiros (1, 9, 5, ...)

```
int hp = 10;
```

- String: Dados de tipo texto representadas entre aspas duplas ("Hello World!")

```
string name = "Player";
```

Float: Dados de tipo numérico com casas decimais (1.5, 3.7, 12.3, ...)

```
float vel = 72.8f;
```

bool: Dados de tipo booleano, verdadeiro ou falso (true or false)

```
bool isDead = false;
```

# Métodos

Os métodos são um conjunto de scripts que só são executados quando o método é chamado, quando isso acontece, todos os scripts dentro do método são executados de uma vez. Para declarar um método devemos seguir:

```
retorno nome(parâmetros) {  
    Bloco de código  
}
```

O retorno é um valor que é passado quando o método é chamado e executado, já os parâmetros são como variáveis, são declaradas junto com a declaração do método assim, quando o método for chamado é preciso passar um valor para cada parâmetro que tiver no método obrigatoriamente para chamá-lo

```
void Movement() {  
    Bloco de código  
}
```

Neste exemplo, o método não possui nenhum retorno, é um retorno vazio (void) e nenhum parâmetro declarado entre os parênteses ( ), ou seja, quando o método for chamado o bloco de código será executado e nada a mais

# Estruturas condicionais

As estruturas condicionais proporcionam dois caminhos para o código, eles verificam se uma sentença é verdadeira ou falsa e dependendo da resposta executam um bloco de código, um se a resposta for verdadeira e um se a resposta for falsa. Para declarar uma condição, devemos fazer:

```
Se(teste) {  
    Bloco de código verdadeiro  
} Senão {  
    Bloco de código falso  
}
```

Existem dois blocos de código e uma verificação de sentença, isto é, se a sentença for verdadeira o primeiro bloco é será executado e o outro não, se for falso, o segundo bloco será executado e o primeiro não, por exemplo:

```
if(8 > 5) {  
    Bloco de código verdadeiro  
} else {  
    Bloco de código falso  
}
```

"8 é maior que 5? Se sim ... Se não ..."

# Arrays, variáveis compostas

Os arrays são variáveis que armazenam vários valores que possuem um índice numérico. Para declarar um array, siga;

```
Tipo[] nome = New tipo[] {valores  
separados por vírgula};
```

Os arrays podem ou não receber um valor inicial assim como variáveis, por exemplo:

```
int[] numeros = New int[] {1, 2, 3, 4, 5, 6,  
7, 8, 9, 10};
```

## Operadores relacionais

Os operadores relacionais comparam dois valores e retornam um valor sendo verdadeiro ou falso. São eles:

>	Maior	8 > 5	True
<	Menor	8 < 5	False
>=	Maior ou igual	8 >= 5	True
<=	Menor ou igual	8 <= 5	False
==	Igual	8 == 5	False
!=	Diferente	8 != 5	True

módulo 5

# Processo de idealização de um jogo

Aprenda a como começar a desenvolver um  
jogo

# Os primeiros passos para a criação de um jogo

Antes de começar a criar um jogo é de extrema importância que você defina o gênero de jogo do seu projeto, isso já te concede uma visão de como o seu jogo será. Entre os muitos gêneros de jogos, temos:

- ação: Envolve desafios de combate e exige coordenação motora e reflexos rápidos
- Aventura: Foca na exploração, solução de problemas e na história
- RPG (Role-Playing Game): Os jogadores assumem o papel de personagens em um mundo fictício
- Simulação: Simula aspectos da vida real ou de outras situações
- Estratégia: Requer planejamento e táticas para vencer
- Esportes: Simula esportes reais como futebol, basquete, entre outros

- Corrida: Foca em competições de velocidade
- Quebra-cabeça: Envolve a resolução de problemas e desafios lógicos
- Tiro em primeira pessoa (FPS): O jogador vê a ação através dos olhos do personagem
- MOBA (Multiplayer Online Battle Arena): Jogos de batalha em arenas online

Além disso, existem também os subgêneros, que dia basicamente a mistura de dois gêneros de jogo, como o subgênero Metroidvania, que combina os elementos das séries de jogos Metroid e Castlevania, sendo um ação-aventura, conhecidos por apresentarem um grande mapa interconectado que permitem o jogador explorar livremente, embora possa haver áreas que necessitam de certos itens e habilidades para serem acessadas, um bom representante desse gênero é o jogo Hollow Knight, da Team Cherry



# Definido a câmera do jogo

O próximo passo é definir o tipo de câmera do seu jogo, como o jogador vai ver o cenário, como será a visão dele em relação ao cenário, neste caso podemos ter dois tipos de câmera, a 2D, quem mostra o mundo em uma dimensão, e a 3D que mostra o jogo em uma visão de 360° graus do cenário

Além disso, dentro da categoria de câmera, tanto 3D como 2D, temos os tipos de visão de jogo, sendo elas:

- **Primeira pessoa:** Simulam a nossa visão real, nos olhos do personagem, não podemos ver seu rosto mas, geralmente, podemos ver do tronco para baixo, muito utilizado em jogos de terror e FPS



- **Terceira pessoa:** Mostra o personagem por completo e propõem uma visão mais ampla do mapa, geralmente é uma câmera que fica nas costas do personagem





- Top Down: Propõem uma visão de cima para baixo do jogo, permitindo enxergar grande parte da área, inimigos e perigos ao redor, também possui variações, como a visão isométrica, que mistura uma visão Top Down com Terceira pessoa, muito utilizada em jogos de masmorra.



- Plataforma 2D: Essa câmera também é utilizada em jogos 3D, mas é geralmente encontrada em jogos 2D, pois a cena é mostrada de forma lateral, proporcionando uma visão ótima de cima, baixo, esquerda e direita, são usadas muito também em jogos Metroidvania



## Criando o protagonista

O próximo passo é criar o nosso personagem principal, para controlarmos do começo ao fim do jogo, isso só se encaixa em alguns tipos de jogo, em jogos de quebra-cabeça, por exemplo, grande parte deles não precisam de um personagem principal, pois é sobre resolver enigmas e problemas, assim como em jogos de FPS, que é um personagem editável, entre outros, mas em jogos do gênero aventura e RPG por exemplo, é necessário sim criar um personagem principal, e para isso, existem algumas dicas que podem ser seguidas:

- Crie uma ficha técnica:
  - Funções
  - Características
  - Acessórios
  - Detalhes únicos
  
- Busque por referências: procure artes que se parecem com o que você está pensando em atribuir ao seu personagem, não precisa necessariamente ser um personagem inteiro que se parece, mas sim artes dos acessórios e características corporais que você possa juntar para formar o seu personagem
  
- Desenhe o esboço em um papel quantas vezes forem necessárias até chegar em um resultado interessante para você
  
- Faça a produção digital e adicione detalhes extras para finalizar a sua arte

Usando essa mesma técnica, você também pode começar a criar as artes dos chefões, inimigos e NPC's do seu jogo

# Estruturando uma mecânica

Agora, com o personagem em mãos, você pode começar a estruturar uma mecânica principal para seu jogo, crie uma ação que seu personagem executará frequentemente, andar, correr, voar, atacar, defender, algo que será muito utilizado no jogo, anote como essa mecânica funcionará e como o seu personagem se comportará com essa mecânica, ele executará uma animação? Uma ação diferente? Essa mecânica terá interações com o cenário em volta? O que é preciso para executar esse a mecânica? Clicar em um botão? Ter um item?



## Definindo um escopo

Uma das partes mais importantes é a definição do escopo do seu jogo, se você nunca fez um jogo é de extrema importância que você crie um jogo de escopo pequeno, não faça um escopo enorme sem ter experiência, se você é iniciante na área, faça pequenos jogos, eventos para a criação de jogos rápidos, como as game Jams, que dia basicamente um desafio a comunidade para criar um jogo em um período curto de tempo, adquira experiência antes de partir para um projeto de escopo maior, isso te trará a experiência necessária e fará uma preparação para você criar seu jogo. Faça jogos de fases curtas, com uma quantidade pequena de fases e publique-o para que a comunidade veja, como na plataforma do itch.io, uma plataforma que possui diversos jogos feitos por uma comunidade de pessoas, ou na própria Playstore, por um preço inicial para publicar seus jogos na plataforma

# Crie um GDD inicial do seu jogo

Crie o GDD e defina todos os pontos que julgar importante, colocando o título do jogo, o gênero, público-alvo, número de jogadores, visão geral do jogo, entre outros, aproveite para criar a história do jogo, se houver uma, veja esse modelo:

- **Título:** A Jornada dos Heróis
- **Gênero:** RPG de Ação e Aventura
- **Público alvo:** Pessoas de 16 ~ 30 anos
  
- **Resumo do jogo:** O jogo começa com o protagonista, Arin, um jovem guerreiro que descobre ser o último descendente de uma antiga linhagem de guardiões. Junto com seus aliados – Lira, uma maga poderosa; Thorne, um ladrão ágil; e Kael, um guerreiro exilado – Arin deve encontrar os fragmentos de um artefato lendário que pode selar a magia caótica e trazer equilíbrio ao mundo
  
- **Visão geral do jogo:** Este jogo é um roguelite em tempo real onde tudo que o jogador precisa entender os horários e percursos dos npcs, descobrir segredos para chantageá-los e manipular os outros personagens do jogo para tentar mudar o seu destino e talvez o destino de

todo o reinado, o tempo não pára em nenhum momento, quando o jogador morrer ele vai voltar ao começo do ciclo e ter novamente as oportunidades que ele perdeu, o jogo conta com diversos finais possíveis

- Referências: ...

## Criando o Level Design

De acordo com a sua experiência, comece a criar o Level Design de uma fase, crie um objetivo que o jogador deve atingir, espalhe obstáculos e inimigos pela fase, crie caminhos secundários e coloque recompensas pela exploração da fase, tudo o que for entreter o jogador, e lembre-se de utilizar o design micro e macro nas fases, claro que isso varia de acordo com o tipo do seu jogo, mas crie um cenário inicial para o seu jogo, apenas tente não criar um nível muito complexo por hora

## As 7 etapas do desenvolvimento

Em certo ponto do desenvolvimento do seu jogo você começará a entrar no que podemos chamar de "as 7 etapas do desenvolvimento", consiste em basicamente fases do desenvolvimento do seu jogo, sendo elas:

**Habilidades:** Você começará a pensar sobre precisar de uma equipe além de você ou trabalhar sozinho no jogo inteiro

Análise de mercado: Será feita uma análise de mercado para saber qual ou quais jogos estão de destacando no momento ou se destacarão

Ideias: Como será o jogo? Estilo, plataformas que será lançado, criar GDD, prazos e metas

Produção: Se iniciará o processo para Criar o jogo, enviar para testes, ajustar com base nos feedbacks, versões alfa, beta, entre outros

Polimento: O jogo entrará em fazer de Finalização, corrigir bugs, refinar os detalhes...

Publicação: O jogo será Lançado nas plataformas, será dado mais foco no marketing, criação do material de divulgação, entre outros

Pós-produção: período em que você irá trabalhar no jogo após o lançamento, lançando patches de correções e aprimoramentos, expansões, entre outros

Grande parte dos jogos que são desenvolvidos, principalmente INDIES, seguem essas etapas, ainda mais provável se for a primeira vez que um estúdio desenvolvendo um jogo

# O resto é com você !

A partir daqui é com você, faça o seu jogo acontecer, crie mecânicas únicas, idealize um jogo de sucesso ! Reúna uma equipe para desenvolver o jogo, escolha uma boa game engine para usar, siga os passos e estude mais sobre se quiser se aprofundar na área ! Boa sorte Game Designer !