Desenvolvimento com Motores de Jogos I

Aula 02 - Implementando Recursos de Jogos

Apresentação

Olá, pessoal! Chegamos à segunda aula da nossa disciplina! Gostaram do que já começamos a fazer? Em apenas uma aula foi possível criar algo que se move, dentro de um dos motores de jogos mais utilizados no mercado. Muito bom!

Na nossa segunda aula, abordaremos um outro assunto bem interessante: recursos de jogos. Há algumas discordâncias em relação à ordem que devem ser passados os conteúdos quando estamos aprendendo um motor novo, como em nosso caso. Algumas pessoas preferem que sejam logo apresentadas as funcionalidades necessárias para que aquela bolinha na plataforma possa fazer coisas diferentes, enquanto outras preferem parar logo no começo e discutir um pouco sobre recursos de jogos para que, assim, possam deixar o jogo mais ou menos com o formato desejado ao final de seu desenvolvimento.

Discutiremos, nesta aula, alguns diferentes aspectos dos recursos de jogos, como o que eles são, de onde vêm ou algo assim. Veremos também o que são os Materiais no Unity e como utilizar o único elemento 2D que o Unity disponibiliza em seu menu - os sprites. Com esses componentes, criaremos um cenário, um jogador e muito mais! Então, vamos lá, antes que recebamos algum chamado do Rob64 (eita... quem lembra desse? É velho, hein!).

Objetivos

Ao final desta aula, você deverá ser capaz de:

- Entender o que são os recursos de jogos e como eles são implementados no Unity;
- Criar e utilizar adequadamente Sprites;
- Criar cenários e personagens utilizando os recursos importados.

O que são os Recursos ou Assets?

Como já discutimos anteriormente, todo jogo é composto por diversos elementos, os quais, ao serem combinados, geram o jogo que vemos. Temos scripts com comportamentos variados, os quais devem ser obedecidos pelo jogo; temos imagens do background das fases, das personagens, dos inimigos, dos obstáculos; temos também os sons que compõem a trilha sonora e os efeitos sonoros; entre outros exemplos. Todos esses elementos utilizados para criar o jogo são conhecidos como recursos, ou assets.

Na aula passada, vimos a Asset Store do Unity e os diversos tipos de elementos que podemos encontrar nela. A grande vantagem de se adquirir esses assets pela loja é o fato de eles já virem em um formato adequado e preparado para que sejam importados no Unity e utilizados pelo desenvolvedor. Ao buscá-los em outras fontes, precisamos sempre atentar para esses detalhes em relação ao formato e ao tipo de asset que queremos incorporar no jogo.

Resumindo, um asset é um objeto qualquer que pode ser utilizado como recurso no seu jogo. Esses arquivos podem ser externos ao Unity, desde que sejam de qualquer formato suportado por este. No entanto, há, ainda, a possibilidade dos assets serem criados dentro do próprio Unity, como os materiais, ou outros elementos que veremos em aulas adiante, como os controladores de animação.

Percebam que o Unity pode receber diversos tipos de arquivos 2D e 3D para serem utilizados como recursos, mas também há a possibilidade, como fizemos na aula passada, de utilizar as primitivas 3D que o Unity traz para substituir temporariamente algum objeto sem qualquer recurso capaz de representá-lo no momento. Caso seja necessário, fique à vontade para utilizar essas formas e substituir os objetos ainda sem recursos do seu jogo por elas. É uma outra forma de avançar no desenvolvimento, mesmo com o atraso na entrega daquele seu amigo artista que está sempre ocupado demais!

Para o nosso jogo exemplo, utilizaremos um asset que criamos aqui mesmo, com os profissionais do Setor de Produção Multimídia do IMD! Caso queiram utilizar um outro asset, podem procurar pela Internet afora ou até mesmo criar os seus

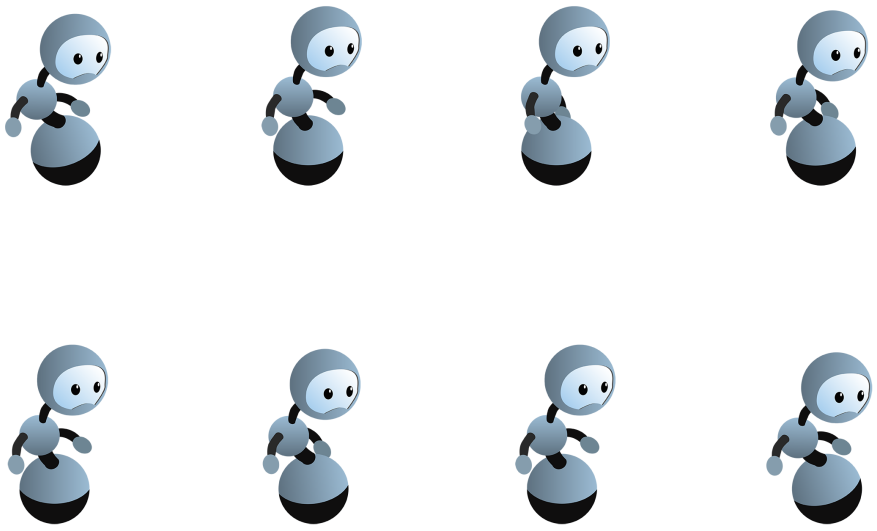
próprios, como veremos na disciplina de Modelagem 2D. Só atentem para o formato, pois é muito importante que os assets estejam num formato aceito pelo Unity. Aos que não quiserem se aventurar com isso no momento, disponibilizamos os assets criados pelo nosso time neste link. Disponibilizaremos cada asset à medida que ele se fizer necessário. Fiquem à vontade para baixar e utilizar! Explicaremos como fazer isso em breve. Antes, vamos falar um pouco sobre sprites.

O que é um Sprite?

Sprite é a marca de um refrigerante com sabor de Lima-limão produzido pela *The Coca-Cola Company* e que está presente em mais de 190 países. O nome sprite significa fada, duendes e espíritos. Não, espera. Acho que cliquei na Wikipedia errada. Vamos tentar de novo!

Wikipedia: Sprites - computação gráfica. Ok! Acho essa explicação melhor! Mas, como geralmente os artigos da Wikipedia sobre assuntos mais técnicos não são muito confiáveis, deixa para lá! Procuraremos uma fonte mais segura, não acha? Então, vamos lá! Ou melhor, deixaremos essa parte mais técnica da discussão para a disciplina de Modelagem 2D, na qual vocês estarão em contato com experts na área. Nós, meros mortais, apenas acompanharemos a definição do próprio Unity e entender que sprites são elementos gráficos 2D utilizados para construir cenas desse tipo. Parece bem simples... E na verdade é! Os sprites são praticamente a base para a construção de todo o nosso jogo 2D. Utilizaremos um sprite para representar o personagem, os inimigos, basicamente tudo! Por isso que o Unity, na parte de objetos 2D, apresenta somente o Sprite como opção de criação! Trata-se de um componente extremamente versátil, empregado a fim de resolver diversos problemas durante a criação do nosso jogo. Vejamos na **Figura 1** um exemplo de vários sprites combinados para o nosso personagem.

Figura 01 - Spritesheet do personagem do nosso jogo exemplo.

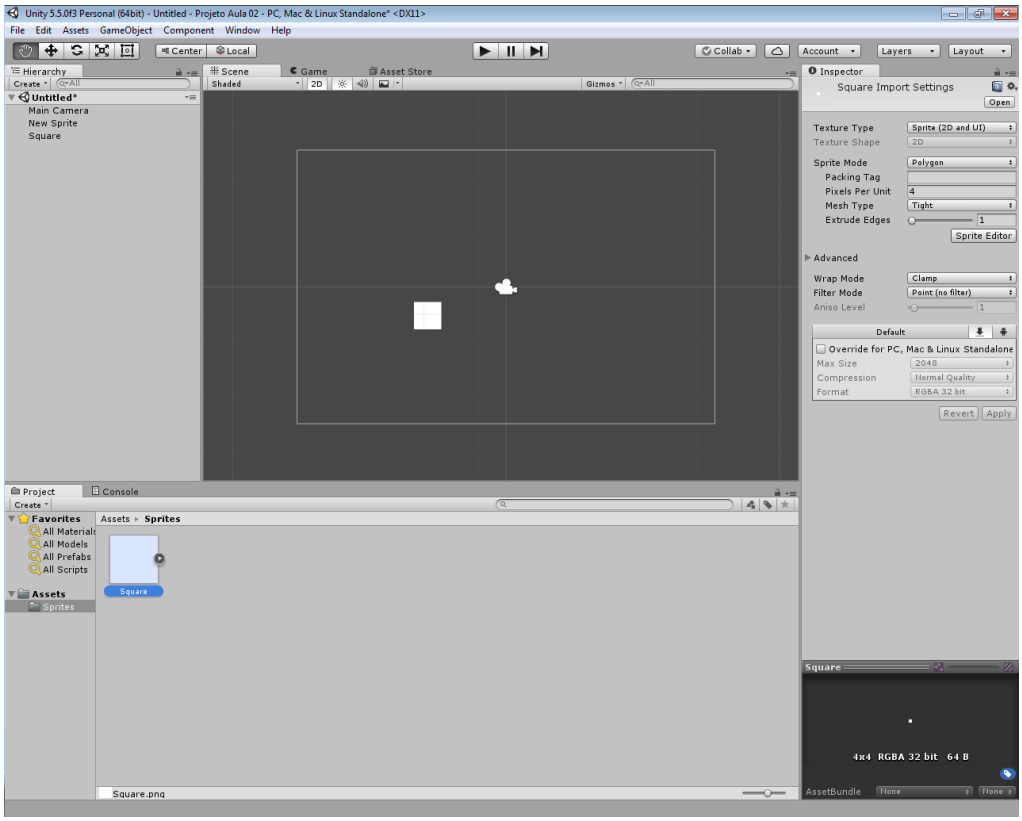


Fonte: Autoria própria (2017).

Interessante... Mas por que são vários sprites posicionados dessa maneira? Novamente, deixemos a discussão para os profissionais de Modelagem 2D, porém apresentaremos aqui o conceito! Trata-se de uma spritesheet. Uma spritesheet contém vários sprites do mesmo objeto de maneira que, se colocados em sequência, um após o outro, gerarão uma animação, como se fossem aqueles desenhos de canto de caderno que vemos por aí, em que as páginas passadas criam a sensação de movimento. Discutiremos animação com mais detalhes em outras aulas, mas é importante que vocês já conheçam esse conceito!

Elementos importantes que também transmitem a ideia de múltiplos sprites em uma mesma imagem são os atlas. Atlas são um conjunto de sprites posicionados na mesma imagem de modo que, quando recortados, formam sprites os quais podem ser utilizados individualmente e separadamente. Veremos mais à frente como o Unity pode automatizar isso. E na disciplina de Modelagem 2D os profissionais entrarão em detalhes! :-P

Figura 02 - Inspector contendo propriedades do sprite selecionado.



Fonte: Captura de tela do Unity. Disponível em: https://unity3d.com/pt/

Atenção!

Até agora vimos os conceitos de assets, sprites, spritesheets e atlas. Se esses conceitos não foram bem assimilados ou você ainda tem dúvida no que são, leia novamente, procure outras fontes, utilize o fórum ou o que você preferir! Mas não siga adiante com dúvidas nesses aspectos.

Atividade 01

1. Que tal elaborar um conceito próprio acerca do tema sprites? Pense em suas aplicações e como estes se encaixam (ou não) na categoria de Assets.

Utilização de Sprites no Unity

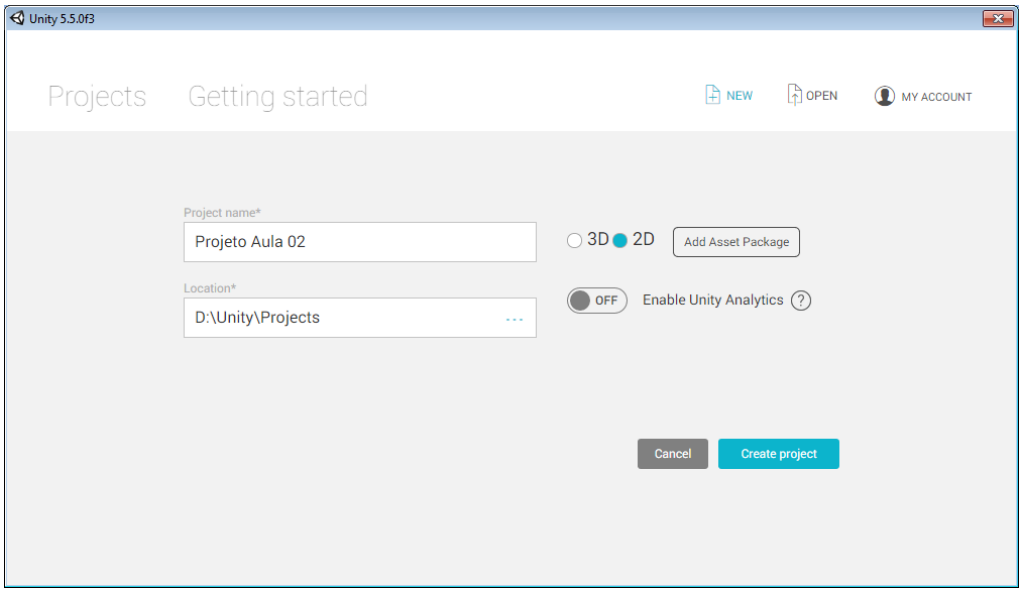
Agora que já entendemos direitinho o que são os assets, e mais especificamente os sprites, discutiremos um pouco como o Unity trata esses elementos.

No Unity, temos um elemento específico dedicado aos sprites, como vimos anteriormente, na parte de GameObjects -> 2D Objects. Esse objeto nada mais é do que um GameObject contendo um Transform e um Sprite Renderer. Esse componente renderer é responsável por exibir o sprite selecionado para aquele objeto e é através dele que o elemento terá alguma imagem. Caso nenhum sprite seja atrelado ao renderer, como é o padrão ao criarmos um novo sprite, o elemento simplesmente não aparecerá na tela, pois não há nada a ser mostrado. Perceba que mesmo assim o sprite existirá - apenas não será mostrado. Criaremos agora um novo sprite para entender melhor os componentes que estão presentes em um elemento desse tipo e discutir as propriedades do componente específico dos sprites.

Criando um Novo Projeto com Sprites

Primeiramente, iniciaremos o Unity e criaremos um novo projeto. Isso já foi abordado na aula anterior e recomendo que você volte lá caso ainda não esteja seguro! Não custa nada *upar* um pouquinho mais na primeira área antes de ir para a próxima! Lembrem-se de configurar o projeto para que seja um projeto 2D! O nome que utilizarei para esse projeto será: Projeto Aula 02! Veja na imagem a seguir, caso tenha alguma dúvida:

Figura 03 - Criação de um projeto no Unity.

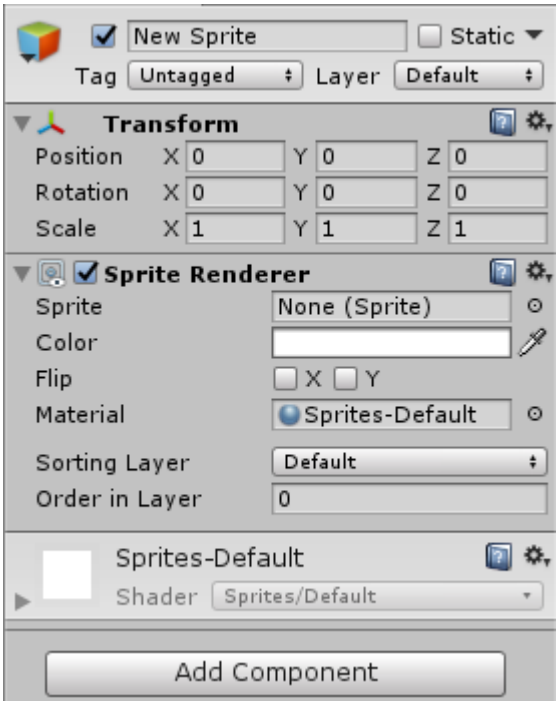


Fonte: Captura de tela do Unity. Disponível em: https://unity3d.com/pt/

Após criar o projeto, adicionaremos um novo objeto do tipo sprite ao nosso projeto vazio. Para isso, utilizaremos o menu GameObject -> 2D Object, no qual você pode escolher qualquer uma das opções que aparecer! Eu escolhi a opção Sprite.

Feito isso, teremos um novo objeto em nossa cena chamado New Sprite. Esse objeto é, como dito anteriormente, um sprite vazio. Ele está em nossa cena, mas simplesmente não aparece por não ter nenhum sprite atrelado a ele. Clicando nele, nós veremos que há alguns componentes presentes. Vejamos esses componentes na **Figura 4**.

Figura 04 - Componentes presentes em um sprite criado pelo menu GameObject.



Fonte: Captura de tela do Unity. Disponível em: https://unity3d.com/pt/

Como já vimos, o primeiro componente presente em nosso sprite é o componente Transform. Esse componente, visto na aula passada, é padrão a todos os objetos e funcionará exatamente da mesma maneira que havíamos visto para os objetos 3D, exceto por um detalhe: não é possível escalar sprites no eixo Z, uma vez que esses objetos são 2D. Perceba, no entanto, que é possível mover e rotacionar esses objetos nos 3 eixos, pois o mundo em que estão inseridos é 3D (se você clicar no botão 2D, que pode ser visto abaixo da aba Game, perceberá isso).

O segundo componente é o componente que torna o sprite o que ele é - o Sprite Renderer. Esse componente é o responsável por exibir, ou renderizar, o sprite na tela. Para tanto, ele traz algumas opções de configuração, por meio de suas propriedades. A primeira destas, Sprite, é a mais importante. É nessa propriedade que selecionamos qual o sprite que queremos exibir através desse elemento. Ele é criado com “None” (e por isso não exibe nada) para que possamos substituir pelo Sprite de nossa preferência assim que o tivermos. No momento, não temos sprite algum, então a lista não conterá qualquer sprite. Veremos em breve como importar um sprite para preencher essa lista.

A segunda propriedade do componente é a color. Ela indica a cor que o nosso sprite terá. A propriedade flip indica o giro do sprite em algum eixo. A propriedade material deve ser mantida com Sprites-Default e indica qual é o material que será utilizado para o Unity exibir o objeto. Por fim, temos duas propriedades relacionadas à camada de organização que o Sprite pertence e, também, a sua ordem dentro dessa camada. Isso poderá servir para organizar os sprites na tela, caso seja necessário. As camadas são renderizadas na ordem definida, mas dentro da camada os objetos com menores números são renderizados primeiro, e os com número maior ficam acima deles.

Vistas todas as propriedades de nosso sprite, importaremos um sprite qualquer para que possamos ver algo na tela, uma vez que o sprite com a propriedade Sprite vazia no Sprite Renderer não exibe qualquer imagem, como já discutido.

Importando um Sprite

A fim de importar um sprite para o nosso projeto, utilizaremos o caminho diretamente pela aba Project do nosso Unity. É MUITO importante SEMPRE mantermos a pasta Assets bem organizada dentro de nosso navegador de projeto. Isso no começo pode parecer bobagem, mas à medida que o projeto ganha novos e novos assets, pode se tornar extremamente caótico.

De modo a evitar que isso aconteça, criaremos desde já o hábito de guardar os elementos onde eles pertencem. Dito isso, cliquemos com o botão direito na pasta Assets em nossa aba Project e selecionemos a opção Create -> Folder. Nomeie esse folder como "Sprites". É nele que guardaremos o nosso novo sprite importado.

Para importar o sprite a esse folder, precisamos, primeiro, selecionar o folder na aba Project, clicando nele. Em seguida, clicamos com o botão direito nesse folder e selecionamos a opção **Import New Asset**. O Unity abrirá uma tela de navegação de pastas para que possamos escolher uma imagem qualquer para ser o nosso novo sprite. Faça isso! Escolha uma imagem qualquer que possua uma extensão PNG e selecione-a como seu novo sprite! Feito isso, um novo sprite será criado em sua pasta Sprites. Clicando nele, um menu de opções no Inspector para configurações desse sprite aparecerá, como visto na **Figura 2**.

Caso o seu projeto tenha sido configurado corretamente para 2D, como indicamos no início dessa etapa, o seu sprite já estará configurado corretamente e pronto para ser utilizado. Caso o Texture Type esteja diferente, isso indica que você não configurou corretamente o seu projeto, então recomendo voltar e prestar atenção nisso! Se o Texture Type está em Sprite (2D and UI), tudo certo!

Outra propriedade interessante é a Sprite Mode. Essa propriedade indica ao Unity o tipo de sprite com o qual estamos lidando para esse sprite. Caso o Sprite Mode seja Single, indicamos que esse é um sprite simples, contendo apenas um elemento. Caso seja Multiple, indicamos que se trata de uma spritesheet ou de um **atlas** - uma imagem contendo diferentes partes de um sprite ou até mesmo de um cenário. Somente em sprites do tipo Multiple podemos fazer cortes para utilizar essas partes separadamente, seja para animação ou simplesmente para as diferentes partes de um só elemento. Já no modo Polygon indicamos que se trata de um polígono, sendo possível modificar/alterar o formato do sprite com máscaras (de cor por exemplo) ou cortes, de acordo com o que for necessário para o Sprite Renderer.

Também é interessante notarmos a propriedade Packing Tag. Essa propriedade é utilizada por uma das ferramentas que o Unity disponibiliza para lidar com sprites. Falaremos dela novamente um pouco mais à frente!

Uma vez importado e configurado o sprite, agora podemos utilizá-lo diretamente em nosso sprite renderer, do nosso elemento New Sprite! Para fazer isso, basta selecionar o elemento na Hierarchy, ir até o componente Sprite Renderer e clicar na bolinha que fica ao lado da propriedade Sprite. Ao fazer isso, o Unity nos dará acesso a um browser de assets do próprio projeto, com opções de busca e tudo mais. Como no momento só temos um asset simples, podemos simplesmente vê-lo diretamente nesse browser e selecioná-lo. Ao fazer isso, o nosso objeto New Sprite já passará diretamente a exibir a imagem selecionada como seu Sprite renderizado. E aí? Qual a imagem que você selecionou? Ficou bacana? Posta lá no fórum!

Ufa! Quanta coisa legal que o Unity nos dá para lidar com sprites, não é? E ainda tem mais! O Unity disponibiliza quatro ferramentas para que possamos trabalhar ainda melhor com sprites. Uma delas, o Sprite Renderer, já começamos a discutir! Vamos, a seguir, entrar em detalhes nessas ferramentas!

Wow! Vimos bastante coisa bem importante nessa seção e realmente vale a pena dar uma parada agora, pegar uma água e tentar garantir que todos esses procedimentos foram assimilados com sucesso. Vimos:

- como o Unity lida com Sprites
- como criar um novo sprite no Unity
- como funciona o componente Sprite Renderer
- como Importar Sprites no Unity e os utilizar

Bastante coisa e que utilizaremos ao longo de todo o resto do curso! Vale a pena garantir que tudo isso foi bem fixado, né?!

Ferramentas de Sprites no Unity

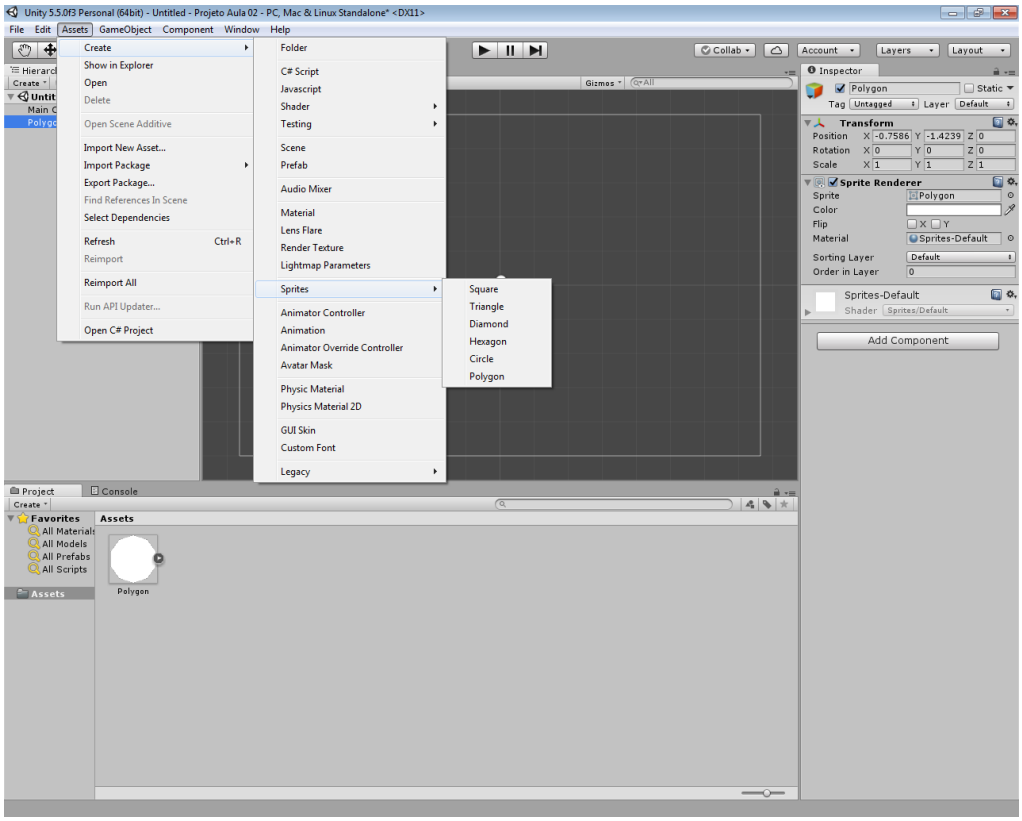
Dada a importância dos sprites para o desenvolvimento de projetos 2D, o Unity nos traz algumas ferramentas para que possamos lidar com sprites de maneira adequada em nosso projeto, sem muita necessidade de auxílio de ferramentas externas, desse modo, cada ferramenta possui um objetivo diferente. Vejamos a seguir as quatro ferramentas disponibilizadas pelo Unity:

Sprite Creator

O Sprite Creator, como o próprio nome diz, serve para criar sprites para o nosso projeto. Trata-se de um menu simples o qual permite que criemos algumas primitivas de sprites a serem utilizadas como placeholders a fim de avançar no desenvolvimento sem que seja necessário esperarmos por uma arte atrasada, ou algo do tipo. A ideia é a mesma das primitivas 3D já citadas, porém, agora em 2D e utilizando o elemento Sprite.

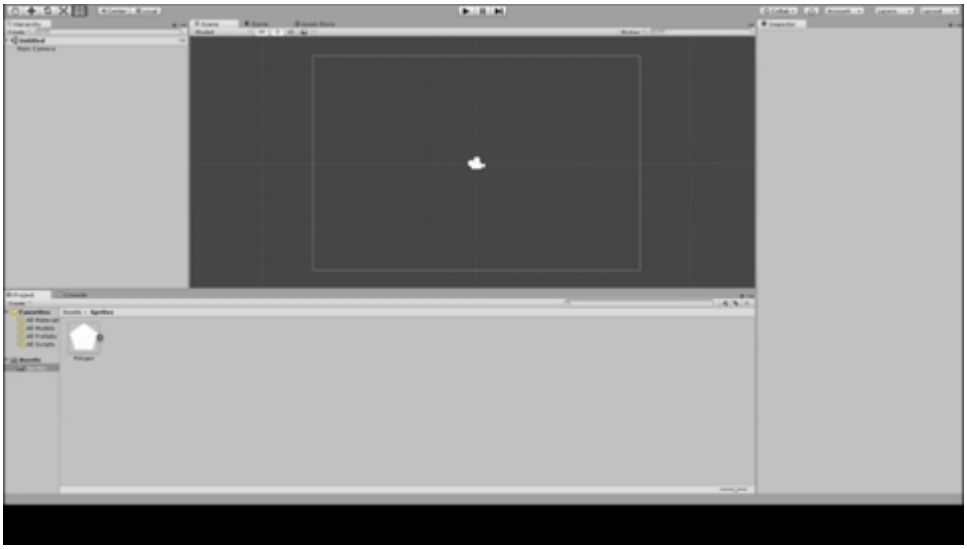
É importante perceber, no entanto, que diferentemente do menu de criação de objetos 3D, o qual cria diretamente um novo objeto em nossa cena, o sprite creator cria um **novo *asset* do tipo sprite** que podemos, então, posicionar em nossa cena. Isso é importante, pois altera um pouco a maneira como adicionamos o nosso objeto criado à cena. Precisamos clicar no asset criado e arrastá-lo até a cena para posicioná-lo pela primeira vez, em vez de simplesmente tê-lo na tela, como acontece com os objetos 3D. Para utilizar a ferramenta, precisamos seguir o menu Assets -> Create -> Sprites, como indicado na **Figura 5**. A adição do sprite à cena, por sua vez, está na **Figura 6**.

Figura 05 - Acessando o Sprite Creator do Unity.



Fonte: Captura de tela do Unity. Disponível em: https://unity3d.com/pt/

Figura 06 - Adicionando o sprite à cena.



Fonte: Captura de tela do Unity. Disponível em: https://unity3d.com/pt/

Veja que os tipos disponíveis para a criação dos sprites são: quadrado, triângulo, diamante, hexágono, círculo e polígono. Essa última forma, no entanto, tem um passo a mais em sua criação. Ao criarmos um polígono, o Unity nos direcionará

automaticamente ao Sprite Editor, o qual veremos a seguir, para podermos alterar, nessa outra ferramenta, a quantidade de lados que o nosso polígono possui.

Após a criação do sprite por meio do menu da **Figura 5**, o Unity o direcionará automaticamente para a pasta onde o Asset foi criado. Caso nenhuma pasta tenha sido selecionada/criada, o sprite aparecerá diretamente na raiz da pasta Assets. Nessa pasta, você terá a opção de alterar o nome do sprite. Você pode utilizar o nome do sprite que aquele placeholder representa ou pode deixar como está, caso queira. É possível renomear os sprites mesmo após ter escolhido o nome, clicando sobre o elemento.

Após criar o objeto e selecionar o nome, ele ficará disponível para utilização em sua pasta, como podemos ver na **Figura 5**, com o elemento chamado Polygon. Agora o objeto está pronto para ser utilizado! Basta clicar nele e arrastá-lo até a nossa Hierarchy para que ele seja criado como um objeto do tipo sprite em nossa cena! Simples, não? A partir daí ele se comportará da mesma maneira que um sprite importado, ou criado pelo menu de GameObject!

Também é possível, uma vez criado esse sprite, utilizá-lo diretamente dentro do Sprite Renderer, pois, como já dissemos, o Sprite Creator cria um novo asset com o sprite escolhido, e não só um objeto. Sendo um asset, podemos utilizá-lo normalmente para o nosso Sprite Renderer.

Bastante útil, não? Essa ferramenta o ajudará no desenvolvimento de jogos diversos, sem que haja a necessidade de perdermos muito tempo com sprites que não temos. Bom! Agora vamos adiante, conhecer mais uma ferramenta importante do Unity para Sprites - o Sprite Editor.

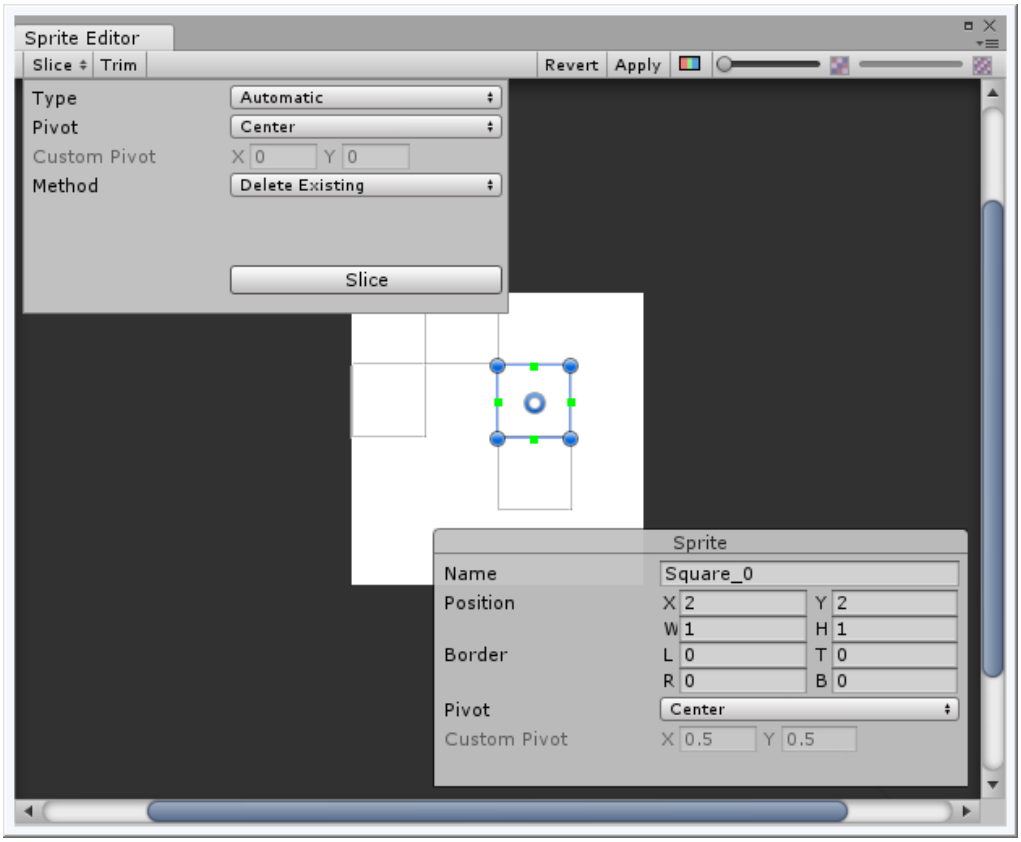
Sprite Editor

Muitas vezes, é comum utilizar técnicas de desenvolvimento para os sprites que envolvem mais de um sprite simultaneamente. Seja através de atlas ou de spritesheets, como já aprendemos, existem alguns casos em que uma só imagem contém mais de um detalhe de um sprite, ou de uma animação. Nesses casos, é importante haver alguma maneira de editarmos esses pedaços para obtermos a nossa imagem principal. Para isso temos o Sprite Editor!

Por meio dessa ferramenta, podemos recortar sprites, separar pedaços para animações, juntar pedaços de sprites, entre outras diversas funcionalidades. Exploraremos um pouco dela agora e entraremos em mais detalhes quando formos utilizar as nossas spritesheets para as animações que veremos adiante no curso.

O primeiro passo para abrirmos o Sprite Editor é garantir que nossos sprites estão configurados adequadamente. Vimos como fazer isso anteriormente, como mostrado na **Figura 2**. Só relembrando, precisamos que o tipo seja ***Sprite (2D and UI)*** e que o Sprite Mode seja ***Multiple***, caso trate-se realmente de um sprite com múltiplos objetos (também podemos trabalhar com sprites únicos). Com tudo configurado adequadamente, ainda conforme a **Figura 2**, temos acesso ao botão Sprite Editor, abaixo das propriedades de Sprite Mode. Ao clicar nesse botão, acessamos o editor de Sprites, como mostrado na **Figura 7**.

Figura 07 - Sprite Editor do Unity.



Fonte: Captura de tela do Unity. Disponível em: https://unity3d.com/pt/

Percebam que essa visualização é para um sprite o qual foi colocado como o modo ***Multiple***. Isso possibilita ativar a parte de "Slice", que permite cortar o sprite em áreas menores para serem trabalhadas. É isso que vemos na parte superior esquerda. Já na parte inferior direita, temos as opções do próprio sprite. Essas

opções podem ser acessadas também nos outros modos e contêm informações sobre posição, borda e o posicionamento do pivô de cada sprite. Nos outros modos, claro, essas opções são reduzidas para ficarem condizentes.

Como falamos anteriormente, mais detalhes sobre o sprite editor serão vistos na aula de animações, uma vez que lá precisaremos cortar, mover e alterar os nossos sprites! Por enquanto, apenas conheçam e aprendam a chegar até ele!

Sprite Renderer

A terceira ferramenta que o Unity nos traz voltada à parte de Sprites é o Sprite Renderer. Essa ferramenta, que é na verdade um componente, é responsável por renderizar os sprites em ambientes 2D e também 3D dentro do Unity.

Como já vimos anteriormente, consistem em seis as propriedades que alteram a maneira a qual esse componente interage com o sprite selecionado para o renderizar.

Citando novamente as seis propriedades, rapidamente:

- **Sprite:** Indica o Sprite a ser renderizado.
- **Color:** Indica a cor da textura renderizada (pode ser especialmente útil em ambientes 3D).
- **Flip:** Indica se deve haver giro do sprite em algum dos eixos indicados.
- **Material:** Material a ser utilizado para a renderização do sprite.
- **Sorting Layer:** Define a camada a qual o Sprite pertence, indicando a ordem de renderização da cena.
- **Order in Layer:** Indica qual a posição do Sprite dentro de sua layer relacionando, assim, sua ordem de renderização.

Um ponto interessante desse componente é justamente essa parte de utilização de camadas. Diferentemente do que acontece em 3D, quando estamos renderizando em 2D temos apenas objetos sobrepondo outros objetos, já que não há a ideia de profundidade. Para que isso ocorra de maneira adequada, é importante que as camadas sejam escolhidas adequadamente, para que os nossos sprites fiquem sempre visíveis quando devem.

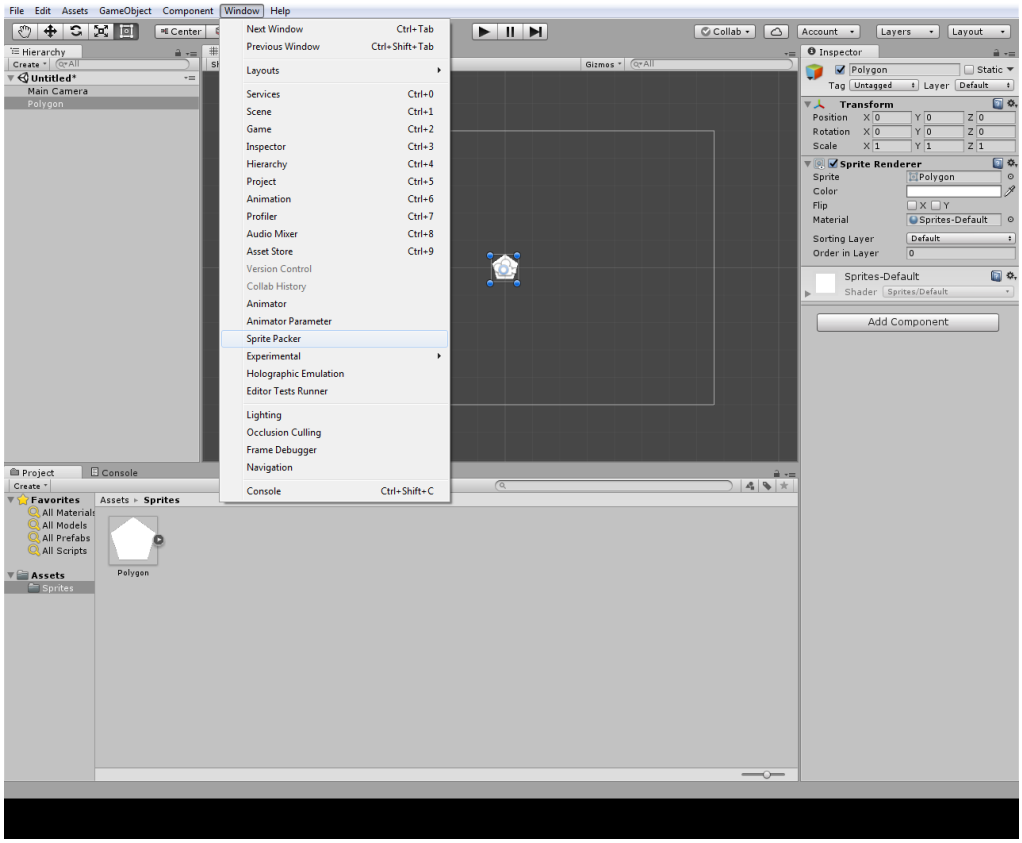
Pense, por exemplo, em um jogo de plataforma clássico, como o Super Mario World. É fácil perceber que há sempre um plano de fundo, que fica por trás de um cenário, o qual está no mesmo plano do Mario. Apesar disso tudo, não há profundidade. São apenas camadas sobrepostas. Alguns outros efeitos, como a paralaxe, dependem também dessa sobreposição adequada de camadas.

Para garantir que tudo isso aconteça de maneira adequada, devemos sempre lembrar do Sorting Layer de nosso sprite, fazendo, assim, com que ele seja renderizado sempre na posição correta em relação ao resto da cena!

Sprite Packer

A quarta e última ferramenta que falaremos nessa aula é o Sprite Packer. Essa ferramenta é disponibilizada pelo Unity para facilitar a criação de diversos atlas para os seus sprites. A **Figura 8** mostra o acesso a esse menu.

Figura 08 - Acessando o Sprite Packer.



Fonte: Captura de tela do Unity. Disponível em: https://unity3d.com/pt/

Basicamente, ao utilizarmos diversos sprites simples, é possível que grandes espaços sejam perdidos devido aos espaços vazios os quais são incluídos em cada uma das texturas, fazendo com que os elementos os quais deveriam ser bem leves se tornem, desnecessariamente, mais pesados. Para evitar esse problema, o Unity disponibiliza o Sprite Packer, o qual recebe diversos sprites simples e os compacta em um atlas, com o menor desperdício de espaço possível, melhorando, assim, o desempenho de seu jogo.

Por padrão, o Sprite Packer não é habilitado. É pouco provável que seja obrigatório, ou extremamente necessário fazer isso com suas texturas, uma vez que raramente se utiliza tanto espaço a partir desses objetos simples. No entanto, fique atento para o fato da possibilidade de que isso seja necessário e é justamente por essa razão que estamos apresentando essa ferramenta. Conhecimento nunca é demais, certo? ;)

Para ativar o Sprite Packer, vá em Edit -> Project Settings -> Editor e, então, escolha se você quer habilitar a ferramenta para “builds” (Enabled for Builds) ou “sempre” (Always Enabled). Caso escolha habilitar para builds, o Unity gerará o Atlas somente quando você compilar uma build do seu projeto, para instalar e testar em algum lugar. Já no modo sempre, ele também gerará os atlas quando você for testar simplesmente pelo modo Play do Unity. O problema disso é que essa geração toma algum tempo, logo, cada vez que você entrar no modo Play, algum tempo a mais será gasto para que haja a compactação.

É possível, também, abrir o Sprite Packer a fim de visualizar a maneira como as texturas estão sendo empacotadas pelo Unity. Para isso, acesse o menu Window -> Sprite Packer e, então, utilize o botão Pack para ver como as texturas se comportarão no seu atlas.

É interessante notar que todo esse processo é feito automaticamente pelo Unity e não é necessário intervir nele diretamente. O único cuidado necessário para que os nossos sprites sejam agrupados adequadamente é o de configurar os sprites os quais devem ser agrupados no nosso Inspector, visto lá atrás, na **Figura 2**. Por meio da propriedade Packing Tag, já estudada, podemos criar grupos de texturas que serão atreladas automaticamente pelo Unity - aquelas com as mesmas tags são colocadas juntas.

Mais detalhes do Sprite Packer podem ser encontrados na documentação oficial do Unity, postado na nossa Leitura Complementar. Qualquer outra referência necessária, ou mais informações que você precise para fazer o seu projeto funcionar, fique à vontade para visitar essa seção!

Atividade 02

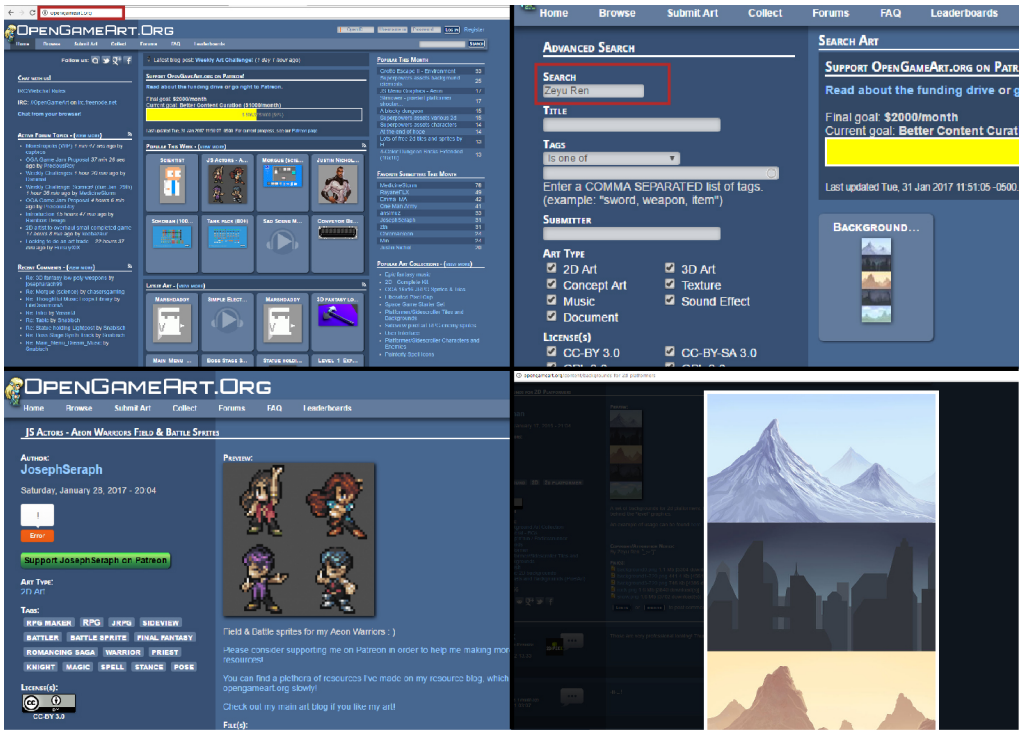
---

1. Cite as quatro principais ferramentas que o Unity disponibiliza para lidar com Sprites e faça um breve resumo da utilidade de cada uma delas.

Adicionando os Primeiros Objetos ao Nosso Jogo

Ok! Então vimos nesta aula diversos conceitos importantes sobre recursos, assets e, principalmente, sprites. Com isso, já podemos, baseados nos materiais que o Setor de Produção Multimídia criou para o nosso jogo, começar a montar uma primeira cena simples, a fim de podermos trabalhar nas próximas aulas, até, finalmente, chegarmos ao nosso jogo completo. Se você quiser acompanhar com os seus próprios assets, fique à vontade! Caso prefira utilizar os nossos, lembro, mais uma vez, que eles serão disponibilizados em links sempre que forem necessários!

No próximo exemplo, também utilizaremos um background especial, obtido a partir de um site que poderá ser MUITO amigo de vocês ao longo da iniciação dessa jornada: o OpenGameArt.org! Especificamente, em nosso primeiro jogo, utilizaremos o arquivo background, criado por Zeyu Ren 任泽宇. Esse arquivo é um dos milhares disponíveis nesse site sob a licença Creative Commons 3.0, a qual permite que você baixe, modifique e utilize livremente, inclusive para fins financeiros, desde que cite o criador e dê um link para a licença! Uma maravilha, não? Agora que já temos os dois sprites necessários, vamos adiante!



Para esse nosso joguinho, vamos começar criando um novo projeto chamado Projeto DJM I. Lembrem-se de configurar o Unity para o modo 2D e, também, selecionar uma pasta conhecida para o projeto, antes de confirmar a criação do novo projeto!

Criado o novo projeto, o primeiro passo para que possamos criar o nosso cenário é, como vimos anteriormente, importar os sprites que utilizaremos para tal. Como já sabemos, devemos sempre manter nossas pastas de recursos organizadas e, para isso, começaremos criando uma nova pasta para os sprites. Em seguida, basta clicar nessa pasta nova e, depois, em Import New Assets. Para começar, importaremos apenas o background e o nosso jogador sem animação. Resumindo:

Atenção!

- Crie uma nova pasta Sprites;
- Botão direito nela -> Import New Assets;
- Selecione o Background e o Player_NoAnim.

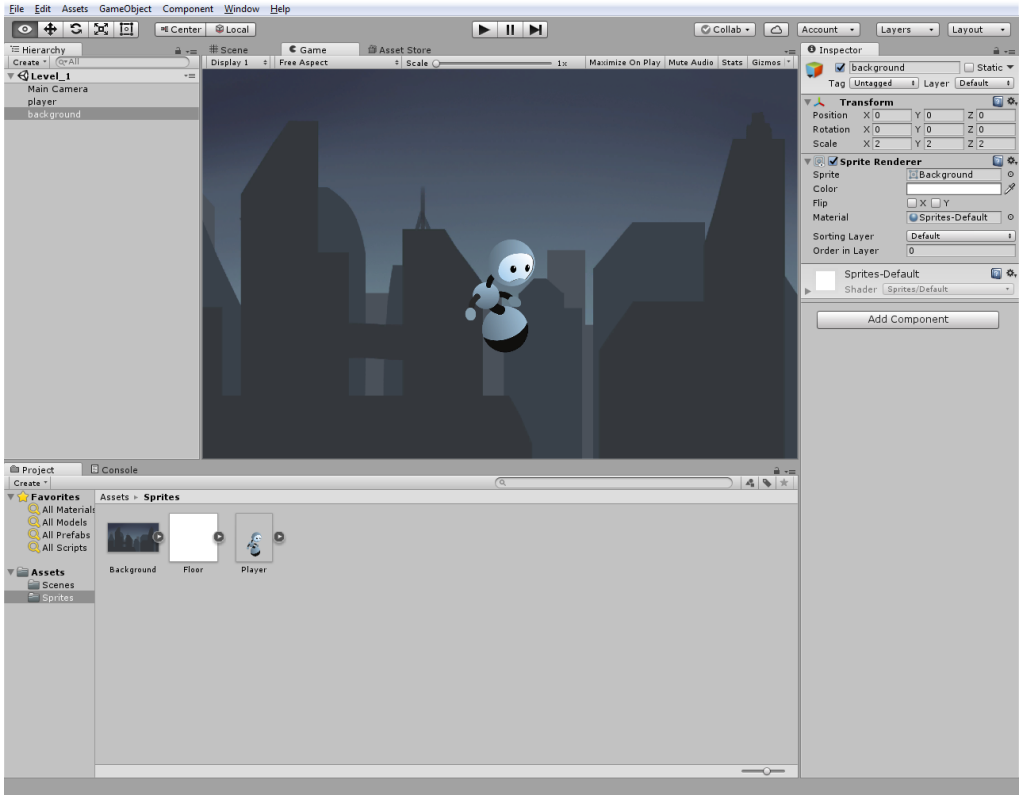
Importados esses dois elementos, basta arrastá-los para que sejam posicionados na cena, como já discutimos anteriormente. O primeiro deles deve ser o background e o segundo, o player. Clique no Asset importado que está dentro da pasta Sprite e arraste-o até a nossa Hierarchy. **Cuidado para não soltar em cima de algum objeto que já está criado!** Isso faria com que o objeto se tornasse filho do objeto sobre o qual foi lançado, resultando na herança de propriedades, entre outras coisas que não pretendemos trabalhar no momento.

Para melhorar a aparência do nosso jogo, faremos algumas modificações após importar os primeiros elementos. Primeiramente, a propriedade Size da nossa Câmera deverá ser alterada: na aba Hierarchy, selecione o nosso objeto **Main Camera** e em seguida, em seu Inspector, altere a propriedade **Size** de 5 para 7. Isso nos dará um maior espaço para trabalhar com os objetos.

Logo após, para que o nosso cenário possa se adequar melhor ao nosso espaço de tela, alteraremos sua escala para 2 em todos os eixos. Já vimos como fazer isso quando discutimos esse componente na aula anterior, lembra? Background -> Inspector -> Scale X = 2, Y = 2 e Z = 2.

Feito isso, já teremos algo parecido com um jogo aparecendo em nossa tela, como vemos na **Figura 9**. Teremos o background ao fundo e o nosso jogador aparecendo à frente dele. Caso isso não esteja acontecendo, modifique a propriedade Order in Layer dos objetos de modo que eles respeitem a ordem de renderização, como estudado. Também se certifique que todos os objetos (exceto a câmera) estão no mesmo eixo Z = 0!

Figura 09 - Primeiros elementos importados em nosso projeto.

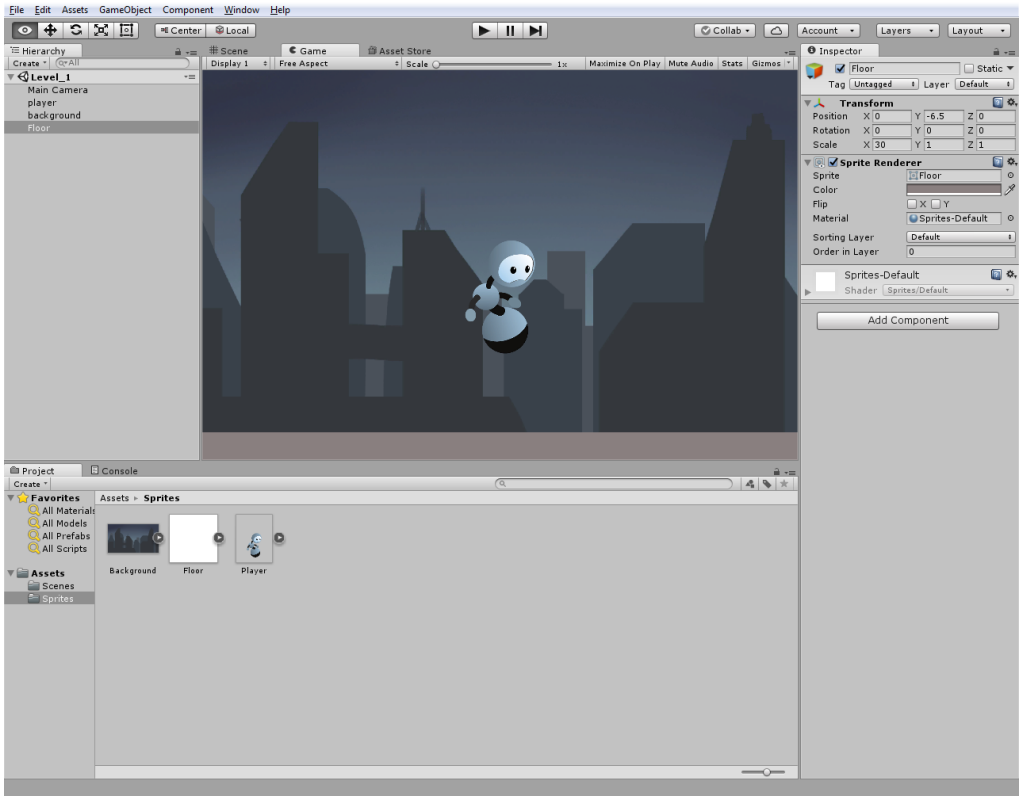


Fonte: Captura de tela do Unity. Disponível em: https://unity3d.com/pt/

Para finalizar esse exemplo, adicionaremos um chão. Nosso chão, para fins de aprendizado, será criado a partir de um sprite vazio, utilizando o Sprite Creator. Lembrando: Assets -> Create -> Sprites -> Square. Nomeiem o quadrado criado como Floor e adicionem-o à pasta Sprites, caso já não tenham criado diretamente

nela. Agora, para que o sprite criado adquira um formato de chão, é necessário alterar algumas propriedades dele! Modifiquem o elemento para que ele fique igual ao exibido na **Figura 10**.

Figura 10 - Cenário inicial criado para o nosso jogo com ajuda do Sprite Creator.



Fonte: Captura de tela do Unity. Disponível em: https://unity3d.com/pt/

Feito isso, precisamos apenas salvar a nossa cena para que ela não se perca em meio às diversas outras coisas que podemos abrir no Unity. Para tanto, utilizamos o atalho mágico que todos os programadores estão acostumados a apertar a cada 5 segundos (ou pelo menos deveriam, antes que percam 30 linhas de código por uma falta de energia!): CTRL + S!

Ao pressionar esse atalho, ou clicar no menu File -> Save Scene, o Unity nos perguntará onde salvar a nossa cena. Lembrando mais uma vez a importância de manter o Assets Folder organizado, criaremos uma nova pasta chamada Scenes e então, dentro dela, salvaremos a nossa cena. O nome pode ser algo como Level_1, por exemplo.

Com isso, concluímos a primeira etapa da criação de nosso jogo - o posicionamento inicial dos objetos que serão utilizados! Agora precisamos seguir adiante e adicionar algum movimento a esses objetos, afinal, o jogo, para ser um jogo, precisa ser jogável! Veremos, na próxima aula, os aspectos básicos de movimentação no Unity. Aprenderemos a controlar o personagem por diferentes métodos de controle e, para que isso seja possível, precisaremos aprender a criar os nossos primeiros scripts em C#! AEEAEEE!! Até lá, povo! o/

Leitura Complementar

Manual do Unity sobre Sprites: https://docs.unity3d.com/Manual/Sprites.html

Um pouco mais sobre Unity 3D: http://producaodejogos.com/fazendo-jogos-e-aplicativos-com-unity-3d/

Resumo

Na aula de hoje, aprendemos conceitos diversos. Começamos entendendo o que são os **recursos** e como são importantes para a criação de nossos jogos. Em seguida, vimos tipos específicos de recursos 2D, que são os **sprites** e suas variações, **spritesheets** e **atlas**.

Além disso, conhecemos a utilização de recursos dentro do Unity e aprendemos como **importar sprites** e **utilizá-los** em nossas cenas do Unity. Na sequência, estudamos quais são as quatro ferramentas que o Unity disponibiliza para facilitar a interação com sprites: **Sprite Creator, Sprite Editor, Sprite Renderer** e **Sprite Packer**. Vimos o funcionamento básico de cada uma delas e o modo como podemos e devemos incorporá-las aos nossos projetos.

Com todo esse conhecimento, **criamos um novo projeto**, ao qual chamamos de Projeto DMJ I, e nele colocamos a mão na massa para **importar, criar e adicionar sprites à nossa cena** inicial. Por fim, vimos como salvar essa cena, lembrando sempre da importância de **manter a nossa pasta de recursos (Assets) organizada!**

Agora que concluímos o projeto da primeira aula, ele ficará disponível como um Unity Package neste link! Caso tenha tido alguma dificuldade, fique à vontade para criar um novo projeto, importar esse Unity Package nele (através do menu Assets -> Import Package -> Custom Package) e verificar o que ficou diferente entre o seu projeto e o que desenvolvemos. Qualquer dúvida, é só passar no fórum!

Autoavaliação

Agora chegou a hora de testar os seus conhecimentos!

Pense no que estudamos até o momento e responda:

1. O que são recursos de jogos? Sprites são os únicos tipos de recursos necessários para jogos 2D?
2. Quais as ferramentas do Unity para lidar com Sprites?
3. Cite os passos para se criar um novo projeto no Unity, importar um sprite qualquer em seu projeto e adicioná-lo à cena principal.

Referências

Documentação oficial do Unity - Disponível em: https://docs.unity3d.com/Manual/index.html.

Tutoriais oficiais do Unity - Disponível em: https://unity3d.com/pt/learn/tutorials

RABIN, Steve. **Introdução ao Desenvolvimento de Games**, Vol 2. CENGAGE.

OpenGameArt - http://opengameart.org/