

Luiz Sergio de Britto Pinto¹ Jucimar Maia da Silva Junior¹

¹Engenharia da Computação
Universidade do Estado do Amazonas - UEA

Dezembro - 2013

Sumário

- Introdução
 - Descrição do Problema
 - Objetivo Geral
 - Objetivos Específicos
- Metodologia
 - Sprites
 - Movimentação
 - Câmera
 - Cenário
 - Inimigos
 - Som de fundo
 - Pontuação
 - Pause
 - Tela Inicial e Final
 - Exportação
- Conclusões

Descrição do Problema

- O desenvolvimento de jogos eletrônicos é uma área que cada vez cresce mais.
- Segundo o jornal da globo, o Brasil é o país onde o mercado de jogos eletrônicos mais cresceu no mundo em 2012.
- Com essa ampliação do mercado de entretenimento virtual, cada vez mais, demanda-se profissionais nesta área que exige conhecimento técnico específico e diferente daquele encontrado em processos de engenharia bem difundidos.

Descrição do Problema

- O desenvolvimento de jogos eletrônicos é uma área que cada vez cresce mais.
- Segundo o jornal da globo, o Brasil é o país onde o mercado de jogos eletrônicos mais cresceu no mundo em 2012.
- Com essa ampliação do mercado de entretenimento virtual, cada vez mais, demanda-se profissionais nesta área que exige conhecimento técnico específico e diferente daquele encontrado em processos de engenharia bem difundidos.

Descrição do Problema

- O desenvolvimento de jogos eletrônicos é uma área que cada vez cresce mais.
- Segundo o jornal da globo, o Brasil é o país onde o mercado de jogos eletrônicos mais cresceu no mundo em 2012.
- Com essa ampliação do mercado de entretenimento virtual, cada vez mais, demanda-se profissionais nesta área que exige conhecimento técnico específico e diferente daquele encontrado em processos de engenharia bem difundidos.

Descrição do Problema

- O processo de desenvolvimento de jogos eletrônicos costuma ser bastante complicado. Além do conhecimento específico necessário, é preciso avaliar as tecnologias disponíveis, requisitos a serem satisfeitos e um conhecimento na área de design para manipular sprites e animações.
- Existem vários frameworks disponíveis para desenvolvimento em várias plataformas diferentes
 - Blender Game Engine
 - Cry Engine
 - UDK
 - Construct
 - Unity

Descrição do Problema

- O processo de desenvolvimento de jogos eletrônicos costuma ser bastante complicado. Além do conhecimento específico necessário, é preciso avaliar as tecnologias disponíveis, requisitos a serem satisfeitos e um conhecimento na área de design para manipular sprites e animações.
- Existem vários frameworks disponíveis para desenvolvimento em várias plataformas diferentes
 - Blender Game Engine
 - Cry Engine
 - UDK
 - Construct
 - Unity

Descrição do Problema

- O processo de desenvolvimento de jogos eletrônicos costuma ser bastante complicado. Além do conhecimento específico necessário, é preciso avaliar as tecnologias disponíveis, requisitos a serem satisfeitos e um conhecimento na área de design para manipular sprites e animações.
- Existem vários frameworks disponíveis para desenvolvimento em várias plataformas diferentes
 - Blender Game Engine
 - Cry Engine
 - UDK
 - Construct
 - Unity

Escolha do Projeto

- Unity uma ótima ferramenta para criação de jogos 3D, porém foi escolhido o desenvolvimento de um jogo plataforma 2D para o trabalho.
- Algumas ferramentas como o Game Maker e o Construct são mais fáceis para iniciantes criarem um jogo 2D, porém não tem tanto poder quanto outras engines maiores.
- Existem várias engines poderosas para a criação de jogos, porém, algumas são bastante difíceis de serem utilizadas para quem nunca usou, como o caso da Blender Game Engine.
- UDK (Unreal Development Kit) e a Cry Engine, além de serem engines muito pesadas, possuem um custo muito alto para funcionalidades medianas e avançadas

Escolha do Projeto

- Unity uma ótima ferramenta para criação de jogos 3D, porém foi escolhido o desenvolvimento de um jogo plataforma 2D para o trabalho.
- Algumas ferramentas como o Game Maker e o Construct são mais fáceis para iniciantes criarem um jogo 2D, porém não tem tanto poder quanto outras engines maiores.
- Existem várias engines poderosas para a criação de jogos, porém, algumas são bastante difíceis de serem utilizadas para quem nunca usou, como o caso da Blender Game Engine.
- UDK (Unreal Development Kit) e a Cry Engine, além de serem engines muito pesadas, possuem um custo muito alto para funcionalidades medianas e avançadas

Escolha do Projeto

- Unity uma ótima ferramenta para criação de jogos 3D, porém foi escolhido o desenvolvimento de um jogo plataforma 2D para o trabalho.
- Algumas ferramentas como o Game Maker e o Construct são mais fáceis para iniciantes criarem um jogo 2D, porém não tem tanto poder quanto outras engines maiores.
- Existem várias engines poderosas para a criação de jogos, porém, algumas são bastante difíceis de serem utilizadas para quem nunca usou, como o caso da Blender Game Engine.
- UDK (Unreal Development Kit) e a Cry Engine, além de serem engines muito pesadas, possuem um custo muito alto para funcionalidades medianas e avançadas

Escolha do Projeto

- Unity uma ótima ferramenta para criação de jogos 3D, porém foi escolhido o desenvolvimento de um jogo plataforma 2D para o trabalho.
- Algumas ferramentas como o Game Maker e o Construct são mais fáceis para iniciantes criarem um jogo 2D, porém não tem tanto poder quanto outras engines maiores.
- Existem várias engines poderosas para a criação de jogos, porém, algumas são bastante difíceis de serem utilizadas para quem nunca usou, como o caso da Blender Game Engine.
- UDK (Unreal Development Kit) e a Cry Engine, além de serem engines muito pesadas, possuem um custo muito alto para funcionalidades medianas e avançadas

Vantagens do Unity

- Mesmo utilizando a versão gratuita do Unity, o mesmo oferece muitos recursos para o desenvolvimento, criação de cenários, etc.
- Permite o desenvolvimento em várias linguagens como Javascript, C# e Boo. Para o projeto foi escolhido a linguagem C# por ter mais documentação e ser uma linguagem mais familiar.
- Permite a exportação do jogo para várias plataformas como Windows, Mac, Android, Iphone, Windows Phone, Xbox, Wii. Para algumas delas, é necessária uma versão específica e paga do Unity.
- É uma ferramenta que está crescendo muito no conceito dos desenvolvedores de jogos e essa seria uma ótima oportunidade de aprender a usar esse framework.

Vantagens do Unity

- Mesmo utilizando a versão gratuita do Unity, o mesmo oferece muitos recursos para o desenvolvimento, criação de cenários, etc.
- Permite o desenvolvimento em várias linguagens como Javascript, C# e Boo. Para o projeto foi escolhido a linguagem C# por ter mais documentação e ser uma linguagem mais familiar.
- Permite a exportação do jogo para várias plataformas como Windows, Mac, Android, Iphone, Windows Phone, Xbox, Wii. Para algumas delas, é necessária uma versão específica e paga do Unity.
- É uma ferramenta que está crescendo muito no conceito dos desenvolvedores de jogos e essa seria uma ótima oportunidade de aprender a usar esse framework.

Vantagens do Unity

- Mesmo utilizando a versão gratuita do Unity, o mesmo oferece muitos recursos para o desenvolvimento, criação de cenários, etc.
- Permite o desenvolvimento em várias linguagens como Javascript, C# e Boo. Para o projeto foi escolhido a linguagem C# por ter mais documentação e ser uma linguagem mais familiar.
- Permite a exportação do jogo para várias plataformas como Windows, Mac, Android, Iphone, Windows Phone, Xbox, Wii. Para algumas delas, é necessária uma versão específica e paga do Unity.
- É uma ferramenta que está crescendo muito no conceito dos desenvolvedores de jogos e essa seria uma ótima oportunidade de aprender a usar esse framework.

Vantagens do Unity

- Mesmo utilizando a versão gratuita do Unity, o mesmo oferece muitos recursos para o desenvolvimento, criação de cenários, etc.
- Permite o desenvolvimento em várias linguagens como Javascript, C# e Boo. Para o projeto foi escolhido a linguagem C# por ter mais documentação e ser uma linguagem mais familiar.
- Permite a exportação do jogo para várias plataformas como Windows, Mac, Android, Iphone, Windows Phone, Xbox, Wii. Para algumas delas, é necessária uma versão específica e paga do Unity.
- É uma ferramenta que está crescendo muito no conceito dos desenvolvedores de jogos e essa seria uma ótima oportunidade de aprender a usar esse framework.

Objetivo Geral

- O trabalho tem por objetivo portar um jogo eletrônico de plataforma 2D feito em flash(NutMeg) para Unity e verificar as vantagens e desvantagens do framework.
- O jogador irá controlar um pássaro que poderá virar e andar a direita, à esquerda e pular sobre as plataformas do jogo. Haverá plataformas fixas e móveis.
- Terá distribuído pelo cenário, estrelas que deverão ser coletadas pelo jogador. Se o número de estrelas coletadas for igual ao número total de estrelas na fase, o jogador vence e passa para a próxima fase, caso tenha.
- Se for a última fase e o jogador coletar todas as estrelas, é mostrada uma tela de vitória. Caso o jogador caia na água ou, seja tocado pelo inimigo que será um gato, o jogador perde e a fase é reiniciada.

Objetivo Geral

- O trabalho tem por objetivo portar um jogo eletrônico de plataforma 2D feito em flash(NutMeg) para Unity e verificar as vantagens e desvantagens do framework.
- O jogador irá controlar um pássaro que poderá virar e andar a direita, à esquerda e pular sobre as plataformas do jogo. Haverá plataformas fixas e móveis.
- Terá distribuído pelo cenário, estrelas que deverão ser coletadas pelo jogador. Se o número de estrelas coletadas for igual ao número total de estrelas na fase, o jogador vence e passa para a próxima fase, caso tenha.
- Se for a última fase e o jogador coletar todas as estrelas, é mostrada uma tela de vitória. Caso o jogador caia na água ou, seja tocado pelo inimigo que será um gato, o jogador perde e a fase é reiniciada.

Objetivo Geral

- O trabalho tem por objetivo portar um jogo eletrônico de plataforma 2D feito em flash(NutMeg) para Unity e verificar as vantagens e desvantagens do framework.
- O jogador irá controlar um pássaro que poderá virar e andar a direita, à esquerda e pular sobre as plataformas do jogo. Haverá plataformas fixas e móveis.
- Terá distribuído pelo cenário, estrelas que deverão ser coletadas pelo jogador. Se o número de estrelas coletadas for igual ao número total de estrelas na fase, o jogador vence e passa para a próxima fase, caso tenha.
- Se for a última fase e o jogador coletar todas as estrelas, é mostrada uma tela de vitória. Caso o jogador caia na água ou, seja tocado pelo inimigo que será um gato, o jogador perde e a fase é reiniciada.

Objetivo Geral

- O trabalho tem por objetivo portar um jogo eletrônico de plataforma 2D feito em flash(NutMeg) para Unity e verificar as vantagens e desvantagens do framework.
- O jogador irá controlar um pássaro que poderá virar e andar a direita, à esquerda e pular sobre as plataformas do jogo. Haverá plataformas fixas e móveis.
- Terá distribuído pelo cenário, estrelas que deverão ser coletadas pelo jogador. Se o número de estrelas coletadas for igual ao número total de estrelas na fase, o jogador vence e passa para a próxima fase, caso tenha.
- Se for a última fase e o jogador coletar todas as estrelas, é mostrada uma tela de vitória. Caso o jogador caia na água ou, seja tocado pelo inimigo que será um gato, o jogador perde e a fase é reiniciada.

Objetivos Específicos

- O jogo terá três fases, sendo a primeira uma fase fácil para o jogador se acostumar com os comandos e o objetivo do jogo, não tendo inimigos para atrapalhar e um mapa pequeno.
- A segunda fase já terá um mapa maior e alguns inimigos para atrapalhar e plataformas que se movem verticalmente.
- A terceira e última fase terá mais inimigos, mais estrelas para serem coletadas em vários lugares e plataformas se movendo horizontalmente e verticalmente.
- O jogo será exportado para um arquivo executável.

Objetivos Específicos

- O jogo terá três fases, sendo a primeira uma fase fácil para o jogador se acostumar com os comandos e o objetivo do jogo, não tendo inimigos para atrapalhar e um mapa pequeno.
- A segunda fase já terá um mapa maior e alguns inimigos para atrapalhar e plataformas que se movem verticalmente.
- A terceira e última fase terá mais inimigos, mais estrelas para serem coletadas em vários lugares e plataformas se movendo horizontalmente e verticalmente.
- O jogo será exportado para um arquivo executável.

Objetivos Específicos

- O jogo terá três fases, sendo a primeira uma fase fácil para o jogador se acostumar com os comandos e o objetivo do jogo, não tendo inimigos para atrapalhar e um mapa pequeno.
- A segunda fase já terá um mapa maior e alguns inimigos para atrapalhar e plataformas que se movem verticalmente.
- A terceira e última fase terá mais inimigos, mais estrelas para serem coletadas em vários lugares e plataformas se movendo horizontalmente e verticalmente.
- O jogo será exportado para um arquivo executável.

Objetivos Específicos

- O jogo terá três fases, sendo a primeira uma fase fácil para o jogador se acostumar com os comandos e o objetivo do jogo, não tendo inimigos para atrapalhar e um mapa pequeno.
- A segunda fase já terá um mapa maior e alguns inimigos para atrapalhar e plataformas que se movem verticalmente.
- A terceira e última fase terá mais inimigos, mais estrelas para serem coletadas em vários lugares e plataformas se movendo horizontalmente e verticalmente.
- O jogo será exportado para um arquivo executável.

- Em computação gráfica, um sprite (do latim spiritus, significando "duende", "fada") é um objeto gráfico bi ou tridimensional que se move numa tela sem deixar traços de sua passagem (como se fosse um "espírito").
- Os sprites foram inventados originalmente como um método rápido de animação de várias imagens agrupadas numa tela, em jogos de computador bidimensionais, usando hardware especial.
- A medida que a performance dos computadores melhorou, esta otimização tornou-se desnecessária e o termo evoluiu para referir-se especificamente às imagens bidimensionais que eram integradas numa determinada cena, isto é, figuras geradas por hardware ou software eram todas referenciadas como sprites.

Sprites

- Em computação gráfica, um sprite (do latim spiritus, significando "duende", "fada") é um objeto gráfico bi ou tridimensional que se move numa tela sem deixar traços de sua passagem (como se fosse um "espírito").
- Os sprites foram inventados originalmente como um método rápido de animação de várias imagens agrupadas numa tela, em jogos de computador bidimensionais, usando hardware especial.
- A medida que a performance dos computadores melhorou, esta otimização tornou-se desnecessária e o termo evoluiu para referir-se especificamente às imagens bidimensionais que eram integradas numa determinada cena, isto é, figuras geradas por hardware ou software eram todas referenciadas como sprites.

Sprites

- Em computação gráfica, um sprite (do latim spiritus, significando "duende", "fada") é um objeto gráfico bi ou tridimensional que se move numa tela sem deixar traços de sua passagem (como se fosse um "espírito").
- Os sprites foram inventados originalmente como um método rápido de animação de várias imagens agrupadas numa tela, em jogos de computador bidimensionais, usando hardware especial.
- A medida que a performance dos computadores melhorou, esta otimização tornou-se desnecessária e o termo evoluiu para referir-se especificamente às imagens bidimensionais que eram integradas numa determinada cena, isto é, figuras geradas por hardware ou software eram todas referenciadas como sprites.

Sprites

- No jogo desenvolvido, para os vários movimentos do personagem principal e movimentos dos inimigos, foi utilizada uma folha de sprites(sprite sheet).
- Essa folha de sprites, contem as várias sprites que podem ser utilizadas. Cada sprite tem exatamente as mesmas dimensões uma da outra, sendo divididas por retângulos.



Figura: Folha de sprites do player



Figura: Folha de sprites do inimigo

Sprites

- No jogo desenvolvido, para os vários movimentos do personagem principal e movimentos dos inimigos, foi utilizada uma folha de sprites(sprite sheet).
- Essa folha de sprites, contem as várias sprites que podem ser utilizadas. Cada sprite tem exatamente as mesmas dimensões uma da outra, sendo divididas por retângulos.



Figura: Folha de sprites do player



Figura: Folha de sprites do inimigo

Sistema de Controle de Sprites

- Existem vários plugins para o Unity que fazem o controle de troca de sprites durante o jogo. Em sua maioria cobram pela utilização.
- Foi implementado um sistema próprio de controle de sprites. Esse sistema foi dividido em várias classes.

Sistema de Controle de Sprites

- Existem vários plugins para o Unity que fazem o controle de troca de sprites durante o jogo. Em sua maioria cobram pela utilização.
- Foi implementado um sistema próprio de controle de sprites. Esse sistema foi dividido em várias classes.

Sistema de Controle de Sprites

- **SpriteRect**: Classe que representa cada retângulo de uma folha de sprites. Essa classe possui as informações sobre as dimensões de cada retângulo dentro da folha de sprites.
- **SpriteSheetInfo**: Classe que guarda informações como nome da sprite, textura a ser utilizada, entre outras informações.
- **Sprite**: Classe responsável pela mudança de uma sprite para outra. Essa classe que controla o tempo que cada sprite deve aparecer ou ser trocada.

Sistema de Controle de Sprites

- **SpriteRect**: Classe que representa cada retângulo de uma folha de sprites. Essa classe possui as informações sobre as dimensões de cada retângulo dentro da folha de sprites.
- **SpriteSheetInfo**: Classe que guarda informações como nome da sprite, textura a ser utilizada, entre outras informações.
- **Sprite**: Classe responsável pela mudança de uma sprite para outra. Essa classe que controla o tempo que cada sprite deve aparecer ou ser trocada.

Sistema de Controle de Sprites

- **SpriteRect**: Classe que representa cada retângulo de uma folha de sprites. Essa classe possui as informações sobre as dimensões de cada retângulo dentro da folha de sprites.
- **SpriteSheetInfo**: Classe que guarda informações como nome da sprite, textura a ser utilizada, entre outras informações.
- **Sprite**: Classe responsável pela mudança de uma sprite para outra. Essa classe que controla o tempo que cada sprite deve aparecer ou ser trocada.

Sistema de Controle de Sprites

- SpriteSet: Classe responsável por chamar o carregamento das sprites. Essa classe é chamada antes de iniciar o jogo e já carrega todas as sprites do jogo.
- XMLParser: Classe que realmente interpreta e carrega as sprites. Todas as informações de sprites são guardadas em um arquivo XML chamado SpriteData.xml. Essa classe de parser, interpreta o xml e todas as informações contidas nele e carrega os sprites passados pelo mesmo.

Sistema de Controle de Sprites

- SpriteSet: Classe responsável por chamar o carregamento das sprites. Essa classe é chamada antes de iniciar o jogo e já carrega todas as sprites do jogo.
- XMLParser: Classe que realmente interpreta e carrega as sprites. Todas as informações de sprites são guardadas em um arquivo XML chamado SpriteData.xml. Essa classe de parser, interpreta o xml e todas as informações contidas nele e carrega os sprites passados pelo mesmo.

Movimentação

- O personagem pode se movimentar para direita, para a esquerda e pular para qualquer um dos lados.
- A cada comando desses, foi necessário implementar uma lógica para, além de mudar as sprites, fazer o jogador realmente se mover.
- Foi utilizado um módulo oferecida pelo próprio Unity, chamado **Character Controller**. Esse controller, te dá várias funcionalidades para controle de movimento e colisões, principalmente em um cenário 2D.

Movimentação

- O personagem pode se movimentar para direita, para a esquerda e pular para qualquer um dos lados.
- A cada comando desses, foi necessário implementar uma lógica para, além de mudar as sprites, fazer o jogador realmente se mover.
- Foi utilizado um módulo oferecida pelo próprio Unity, chamado **Character Controller**. Esse controller, te dá várias funcionalidades para controle de movimento e colisões, principalmente em um cenário 2D.

Movimentação

- O personagem pode se movimentar para direita, para a esquerda e pular para qualquer um dos lados.
- A cada comando desses, foi necessário implementar uma lógica para, além de mudar as sprites, fazer o jogador realmente se mover.
- Foi utilizado um módulo oferecida pelo próprio Unity, chamado **Character Controller**. Esse controller, te dá várias funcionalidades para controle de movimento e colisões, principalmente em um cenário 2D.

Câmera

- A câmera deve sempre seguir o personagem durante seu movimento pelo cenário.
- Existe um jeito bastante simples de fazer isso que o Unity proporciona. Apenas botando o objeto da câmera principal como um objeto filho do personagem, a câmera iria seguir o movimento do personagem.
- Esse método acarreta em alguns problemas como o caso que a câmera acabaria mostrando a parte que não existe do cenário.
- Para resolver esse problema, foi feito um controle simples pelo código que faz com que a câmera acompanhe o personagem até o limite especificado para cada fase.

Câmera

- A câmera deve sempre seguir o personagem durante seu movimento pelo cenário.
- Existe um jeito bastante simples de fazer isso que o Unity proporciona. Apenas botando o objeto da câmera principal como um objeto filho do personagem, a câmera iria seguir o movimento do personagem.
- Esse método acarreta em alguns problemas como o caso que a câmera acabaria mostrando a parte que não existe do cenário.
- Para resolver esse problema, foi feito um controle simples pelo código que faz com que a câmera acompanhe o personagem até o limite especificado para cada fase.

Câmera

- A câmera deve sempre seguir o personagem durante seu movimento pelo cenário.
- Existe um jeito bastante simples de fazer isso que o Unity proporciona. Apenas botando o objeto da câmera principal como um objeto filho do personagem, a câmera iria seguir o movimento do personagem.
- Esse método acarreta em alguns problemas como o caso que a câmera acabaria mostrando a parte que não existe do cenário.
- Para resolver esse problema, foi feito um controle simples pelo código que faz com que a câmera acompanhe o personagem até o limite especificado para cada fase.

Câmera

- A câmera deve sempre seguir o personagem durante seu movimento pelo cenário.
- Existe um jeito bastante simples de fazer isso que o Unity proporciona. Apenas botando o objeto da câmera principal como um objeto filho do personagem, a câmera iria seguir o movimento do personagem.
- Esse método acarreta em alguns problemas como o caso que a câmera acabaria mostrando a parte que não existe do cenário.
- Para resolver esse problema, foi feito um controle simples pelo código que faz com que a câmera acompanhe o personagem até o limite especificado para cada fase.

Cenário

- Na maioria dos jogos, os cenários são criados por outras ferramentas especializadas e importadas para o unity. No caso desse projeto, nenhuma ferramenta especializada foi utilizada.
- A partir de uma página de sprites com várias partes possíveis do cenário, foram criados sprites específicos. Com esses sprites específicos, foi utilizado de outra ferramenta do unity, chamado prefab.
- Um prefab é apenas um modelo de objeto que pode ser instanciado ou destruído em qualquer momento do jogo. Tendo esse modelo, ao iniciar o jogo, é montado o cenário inteiro a partir desses prefabs.
- Utilizar um prefab é um jeito bem fácil e prático para instanciar objetos que se repetem várias vezes no cenário durante o jogo. Todos eles possuem sempre as mesmas propriedades básicas.

Cenário

- Na maioria dos jogos, os cenários são criados por outras ferramentas especializadas e importadas para o unity. No caso desse projeto, nenhuma ferramenta especializada foi utilizada.
- A partir de uma página de sprites com várias partes possíveis do cenário, foram criados sprites específicos. Com esses sprites específicos, foi utilizado de outra ferramenta do unity, chamado prefab.
- Um prefab é apenas um modelo de objeto que pode ser instanciado ou destruído em qualquer momento do jogo. Tendo esse modelo, ao iniciar o jogo, é montado o cenário inteiro a partir desses prefabs.
- Utilizar um prefab é um jeito bem fácil e prático para instanciar objetos que se repetem várias vezes no cenário durante o jogo. Todos eles possuem sempre as mesmas propriedades básicas.

Cenário

- Na maioria dos jogos, os cenários são criados por outras ferramentas especializadas e importadas para o unity. No caso desse projeto, nenhuma ferramenta especializada foi utilizada.
- A partir de uma página de sprites com várias partes possíveis do cenário, foram criados sprites específicos. Com esses sprites específicos, foi utilizado de outra ferramenta do unity, chamado prefab.
- Um prefab é apenas um modelo de objeto que pode ser instanciado ou destruído em qualquer momento do jogo. Tendo esse modelo, ao iniciar o jogo, é montado o cenário inteiro a partir desses prefabs.
- Utilizar um prefab é um jeito bem fácil e prático para instanciar objetos que se repetem várias vezes no cenário durante o jogo. Todos eles possuem sempre as mesmas propriedades básicas.

Cenário

- Na maioria dos jogos, os cenários são criados por outras ferramentas especializadas e importadas para o unity. No caso desse projeto, nenhuma ferramenta especializada foi utilizada.
- A partir de uma página de sprites com várias partes possíveis do cenário, foram criados sprites específicos. Com esses sprites específicos, foi utilizado de outra ferramenta do unity, chamado prefab.
- Um prefab é apenas um modelo de objeto que pode ser instanciado ou destruído em qualquer momento do jogo. Tendo esse modelo, ao iniciar o jogo, é montado o cenário inteiro a partir desses prefabs.
- Utilizar um prefab é um jeito bem fácil e prático para instanciar objetos que se repetem várias vezes no cenário durante o jogo. Todos eles possuem sempre as mesmas propriedades básicas.

Inimigos

- Para a criação dos inimigos, foi utilizado do mesmo sistema de controle de sprites utilizado no controle do personagem principal.
- A diferença é que foi criada uma lógica diferente para a movimentação. Cada inimigo já tem uma rota certa no mapa com um limite máximo a ser percorrido, podendo começar a rota para esquerda ou para a direita, dependendo da configuração.
- Do mesmo jeito que o cenário, a criação dos inimigos é feita por prefabs.
- Foi implementada uma lógica que ao ser detectada uma colisão entre um objeto instanciado de um inimigo e o personagem principal, o jogo considera como derrota e a fase reinicia.

Inimigos

- Para a criação dos inimigos, foi utilizado do mesmo sistema de controle de sprites utilizado no controle do personagem principal.
- A diferença é que foi criada uma lógica diferente para a movimentação. Cada inimigo já tem uma rota certa no mapa com um limite máximo a ser percorrido, podendo começar a rota para esquerda ou para a direita, dependendo da configuração.
- Do mesmo jeito que o cenário, a criação dos inimigos é feita por prefabs.
- Foi implementada uma lógica que ao ser detectada uma colisão entre um objeto instanciado de um inimigo e o personagem principal, o jogo considera como derrota e a fase reinicia.

Inimigos

- Para a criação dos inimigos, foi utilizado do mesmo sistema de controle de sprites utilizado no controle do personagem principal.
- A diferença é que foi criada uma lógica diferente para a movimentação. Cada inimigo já tem uma rota certa no mapa com um limite máximo a ser percorrido, podendo começar a rota para esquerda ou para a direita, dependendo da configuração.
- Do mesmo jeito que o cenário, a criação dos inimigos é feita por prefabs.
- Foi implementada uma lógica que ao ser detectada uma colisão entre um objeto instanciado de um inimigo e o personagem principal, o jogo considera como derrota e a fase reinicia.

Inimigos

- Para a criação dos inimigos, foi utilizado do mesmo sistema de controle de sprites utilizado no controle do personagem principal.
- A diferença é que foi criada uma lógica diferente para a movimentação. Cada inimigo já tem uma rota certa no mapa com um limite máximo a ser percorrido, podendo começar a rota para esquerda ou para a direita, dependendo da configuração.
- Do mesmo jeito que o cenário, a criação dos inimigos é feita por prefabs.
- Foi implementada uma lógica que ao ser detectada uma colisão entre um objeto instanciado de um inimigo e o personagem principal, o jogo considera como derrota e a fase reinicia.

Som de Fundo

- Para o som de fundo, a unity nos proporciona uma funcionalidade chamada de Audio Source, a qual você pode escolher um som e mudar algumas configurações do mesmo.
- Ao criar essa funcionalidade em um objeto vazio, o objeto foi incluído como um objeto filho do personagem principal, desse modo o som iria sempre estar ativo não importando se o player se movesse.

Som de Fundo

- Para o som de fundo, a unity nos proporciona uma funcionalidade chamada de Audio Source, a qual você pode escolher um som e mudar algumas configurações do mesmo.
- Ao criar essa funcionalidade em um objeto vazio, o objeto foi incluído como um objeto filho do personagem principal, desse modo o som iria sempre estar ativo não importando se o player se movesse.

Pontuação

- Durante a criação do cenário, também são criados vários objetos, utilizando do prefab, que são estrelas a ser capturadas pelo personagem.
- Para pontuação do jogo, foi implementado uma lógica na qual a cada colisão do personagem com um objeto estrela, a pontuação aumenta.
- Caso a quantidade de estrelas capturadas fosse o mesmo da quantidade de estrelas totais criadas no cenário, significa que o jogador passou da fase.

Pontuação

- Durante a criação do cenário, também são criados vários objetos, utilizando do prefab, que são estrelas a ser capturadas pelo personagem.
- Para pontuação do jogo, foi implementado uma lógica na qual a cada colisão do personagem com um objeto estrela, a pontuação aumenta.
- Caso a quantidade de estrelas capturadas fosse o mesmo da quantidade de estrelas totais criadas no cenário, significa que o jogador passou da fase.

Pontuação

- Durante a criação do cenário, também são criados vários objetos, utilizando do prefab, que são estrelas a ser capturadas pelo personagem.
- Para pontuação do jogo, foi implementado uma lógica na qual a cada colisão do personagem com um objeto estrela, a pontuação aumenta.
- Caso a quantidade de estrelas capturadas fosse o mesmo da quantidade de estrelas totais criadas no cenário, significa que o jogador passou da fase.

Pause

- Foi criada uma lógica para toda vez que o jogador apertar a tecla ESC, todos os objetos do jogo são congelados e nenhum comando pode ser executado até a próxima vez que o jogador apertar novamente a tecla ESC e o jogo voltaria a rodar de onde parou.

Tela inicial e final

- Ao começar o jogo, foi criada uma tela contendo o nome do jogo e informações para iniciar o mesmo. Ao apertar a tecla enter, essa tela some e o jogo inicia realmente.
- Caso o jogador vença a terceira e última fase, uma tela que informa a vitória do jogador aparece. Nessa última tela a única opção do jogador é apertar a tecla enter e sair do jogo.

Tela inicial e final

- Ao começar o jogo, foi criada uma tela contendo o nome do jogo e informações para iniciar o mesmo. Ao apertar a tecla enter, essa tela some e o jogo inicia realmente.
- Caso o jogador vença a terceira e última fase, uma tela que informa a vitória do jogador aparece. Nessa última tela a única opção do jogador é apertar a tecla enter e sair do jogo.

Exportação do jogo

- O Unity proporciona um jeito fácil para exportar o jogo para várias plataformas. No caso desse jogo, foi realizada apenas a exportação para um arquivo executável(.exe) para PC.

Conclusões

- Unity é uma ferramenta muito poderosa para criação de jogos, porém na versão utilizada, o foco era a criação de jogos 3D.
- Um designer na equipe pode poupar muito tempo do desenvolvedor. A utilização de ferramentas especializadas na criação de cenários e assets é algo que facilita bastante.
- Outras engines para desenvolvimento de jogos podem ser mais efetivas para a criação de um jogo como o proposto pelo projeto, porém não possuem a quantidade de opções e ferramentas que o Unity proporciona.
- A curva de aprendizagem pode ser um pouco elevada para quem nunca criou um jogo, porém após um tempo de estudo, o Unity se torna uma ferramenta muito poderosa.

Conclusões

- Unity é uma ferramenta muito poderosa para criação de jogos, porém na versão utilizada, o foco era a criação de jogos 3D.
- Um designer na equipe pode poupar muito tempo do desenvolvedor. A utilização de ferramentas especializadas na criação de cenários e assets é algo que facilita bastante.
- Outras engines para desenvolvimento de jogos podem ser mais efetivas para a criação de um jogo como o proposto pelo projeto, porém não possuem a quantidade de opções e ferramentas que o Unity proporciona.
- A curva de aprendizagem pode ser um pouco elevada para quem nunca criou um jogo, porém após um tempo de estudo, o Unity se torna uma ferramenta muito poderosa.

Conclusões

- Unity é uma ferramenta muito poderosa para criação de jogos, porém na versão utilizada, o foco era a criação de jogos 3D.
- Um designer na equipe pode poupar muito tempo do desenvolvedor. A utilização de ferramentas especializadas na criação de cenários e assets é algo que facilita bastante.
- Outras engines para desenvolvimento de jogos podem ser mais efetivas para a criação de um jogo como o proposto pelo projeto, porém não possuem a quantidade de opções e ferramentas que o Unity proporciona.
- A curva de aprendizagem pode ser um pouco elevada para quem nunca criou um jogo, porém após um tempo de estudo, o Unity se torna uma ferramenta muito poderosa.

Conclusões

- Unity é uma ferramenta muito poderosa para criação de jogos, porém na versão utilizada, o foco era a criação de jogos 3D.
- Um designer na equipe pode poupar muito tempo do desenvolvedor. A utilização de ferramentas especializadas na criação de cenários e assets é algo que facilita bastante.
- Outras engines para desenvolvimento de jogos podem ser mais efetivas para a criação de um jogo como o proposto pelo projeto, porém não possuem a quantidade de opções e ferramentas que o Unity proporciona.
- A curva de aprendizagem pode ser um pouco elevada para quem nunca criou um jogo, porém após um tempo de estudo, o Unity se torna uma ferramenta muito poderosa.

Referências



[1] Carlos de Lannoy. Disponível em <http://g1.globo.com/jornal-daglobo/noticia/2013/05/brasil-lidera-crescimento-do-mercado-de-jogos-eletronicosem-2012.html> .
Acessado em: 24/09/2013.



[2] Unity. Disponível em <http://portuguese.unity3d.com/gallery/made-withunity/game-list> .
Acessado em: 25/09/2013.