
EPTA ENTERTAINMENT

NESTE DOCUMENTO ENCONTRA-SE O PLANO DE DESENVOLVIMENTO QUE SE REFERE À CRIAÇÃO DA VERSÃO 1.0 DO APLICATIVO EPTA SPACE PROGRAM, DESENVOLVIDO PELA EPTA ENTERTAINMENT.

DESENVOLVIDO POR

FELIPE J. O. RIBEIRO
MATEUS DA SILVA FERNANDES
OLAVO CAETANO INÁCIO
PEDRO GUILHERME R. V. DE MELO

*Universidade Federal de
Uberlândia*

EQUIPE DE PROPULSÃO E TECNOLOGIA
AEROESPACIAL

EPTA SPACE PROGRAM - VERSÃO 1.0

16 DE JANEIRO DE 2021

Resumo

Neste documento encontram-se as práticas adotadas no desenvolvimento da versão 1.0 do aplicativo Epta Space Program. O programa já havia sido lançado na google play em meados de 2017, mas ele foi retirado do ar devido a questões burocráticas (ausência de uma política de privacidade).

Nesta etapa de desenvolvimento a equipe atualizará o jogo para a versão mais recente do unity. Também revisará o work flow uma vez que na versão de 2017 estava impossível de escalar e inviável de manter devido a más práticas de programação. Como resposta a isso, a sub área se propôs a revisar todos os padrões de desenvolvimento de forma a estar conforme com as mais altas recomendações desta indústria.

As funções base do aplicativo são:

- **TELA INICIAL** Desenvolvimento da tela inicial com animações. Que não precise iniciar o jogo pausado.
- **3 FASES** Três etapas distintas de jogo. (céu azul, extratosfera, espaço)
- **FASE FINAL** A certa altura (a determinar), o player tem a opção de de ir para a lua. (o conteúdo será adicionado posteriormente. Mas por hora o player recebe uma mensagem de congratulações)
- **MORTE** Som e efeito visual de explosão no momento de morte.
- **ARCADE** Mecânica base do jogo mantém-se arcade, com rolagem para cima com obstáculos e replay ágil.

As features que espera-se que estejam presentes no jogo na data de estreia são:

- **PROPAGANDA** As propagandas tipo banner e tipo Intersticial devem ser implementadas novamente.
- **VERSÃO PAGA** Deve haver uma versão sem propagandas e que é paga. (preço a determinar)
- **TABELA INFO** Tabela de informações sobre a equipe no menu opções.
- **VOLUME** Deve ser possível ajustar o volume.
- **SALVAR** A melhor pontuação do jogador deve ser salva localmente, assim como o volume.
- **ASSETS REUTILIZADOS** Serão utilizados os assets do antigo lançamento. Novos devem ser feitos conforme necessidade.
- **ALTURAS REAIS** As alturas de mudança de fase devem ser consistentes com o mundo real.
- **DIFICULDADE CRESCENTE** O sistema de dificuldade deve ser implementado de forma a permitir razoável controle em função de fase e tempo.

A metodologia adotada para a criação do APP tem duas etapas importantes: Até o término da versão 1.0 será adotada uma metodologia em cascata, nesta que é a etapa de pré lançamento, onde se dividirão as features a implementar em uma ordem lógica de execução. Datas para pontos chave no desenvolvimento serão determinadas seguindo o padrão industrial (pre-alpha : 0.5.0, alpha : 0.6.0, beta-1 : 0.7.0, beta-2 : 0.8.0, ..., release : 1.0.0), onde cada uma terá um objetivo e uma lista de requerimentos a serem atendidos. Em seguida, após lançado, será adotada a metodologia ágil para desenvolvimento contínuo de atualizações e melhoramentos conforme necessário. Esse será chamada de etapa contínua de pós lançamento.

Será implementado um ciclo de desenvolvimento completamente integrado à ferramenta Git de controle de versão e à página GitHub (https://github.com/Epta-space/EPTA_SPACE_PROGRAM), onde o jogo será salvo em cloud em um repositório privado. Assim, essas ferramentas servirão para possibilitar o desenvolvimento em grupo do código fonte, além de também ser uma forma de quantificação da produtividade, uma vez que o controle do número de commits é uma métrica importante no acompanhamento das atividades. Mais detalhes podem ser encontrados na descrição das etapas de desenvolvimento do pré lançamento.

A criação do produto mínimo viável já foi feita na primeira versão do jogo, motivo pelo qual não temos a etapa MVP neste texto. Apesar disso será inserida aqui alguma documentação sobre esse processo nos capítulos iniciais. O desenvolvimentos que ocorreram anteriores a esse relatório são o motivo, também, da enumeração das etapas começarem em 0.5. Nos capítulos iniciais também será dada uma detalhada noção do estado atual do aplicativo assim como algumas informações sobre como foi o desenvolvimento até então. Tudo que foi feito até o momento foi desenvolvido durante o ano corrido de 2020, com exceção dos assets que vem da versão anterior.

Dentre as informações do estado atual do aplicativo serão listados os scripts que existem atualmente em formato UML e por escrito, a estrutura de arquivos no projeto unity e a estrutura de Prefabs. Estas estruturas também já atendem a padrões de projeto sugeridos online e facilitam o desenvolvimento do controle de versão via Git, além de organizar as informações para facilitar o acesso e aumentar a escalabilidade.

Próximo ao fim do documento também serão apresentadas algumas propostas de melhoramento na etapa ágil de lançamento, isso é, após a primeira versão do programa ser lançada na Google Play. Isso mostra as possíveis rotas que a subárea tomará no futuro, após o começo da etapa de desenvolvimento contínuo.

Palavras chave: Desenvolvimento de Software, aplicativo, android, jogo, google play.

Sumário

1	O desenvolvimento até aqui	1
1.1	Sobre o papel do desenvolvimento de jogos na equipe	2
1.2	O Mínimo produto viável(MVP)	2
2	Estado do aplicativo	3
2.1	Disposição de arquivos	3
2.2	Disposição de Game Objects	3
2.3	Ecossistema lógico	3
3	Plano de gestão para 2021	5
4	Planos futuros	6

Lista de Figuras

1.1	Downloads do aplicativo distribuídos no tempo.	1
2.1	Código do script "Destroy_obstacles.cs".	4
2.2	Código do script "audio_controls.cs".	4

Capítulo 1

O desenvolvimento até aqui

O início deste projeto ocorreu no começo do primeiro semestre de 2018. Um grupo de membros se juntou no desenvolvimento do aplicativo como uma nova proposta de angariação de fundos para a equipe. Observou-se o potencial do mercado de aplicativos global, que cresce consistentemente ao longo dos anos. Além disso pensou-se na oportunidade de se aprofundar no desenvolvimento de software e o aprendizado que isso traria.

Ao longo do ano o aplicativo foi desenvolvido e finalmente lançado em dezembro de 2018. Ele foi baixado em mais de 40 países e mais de 800 vezes, com uma avaliação média de 4.88 estrelas de 5.

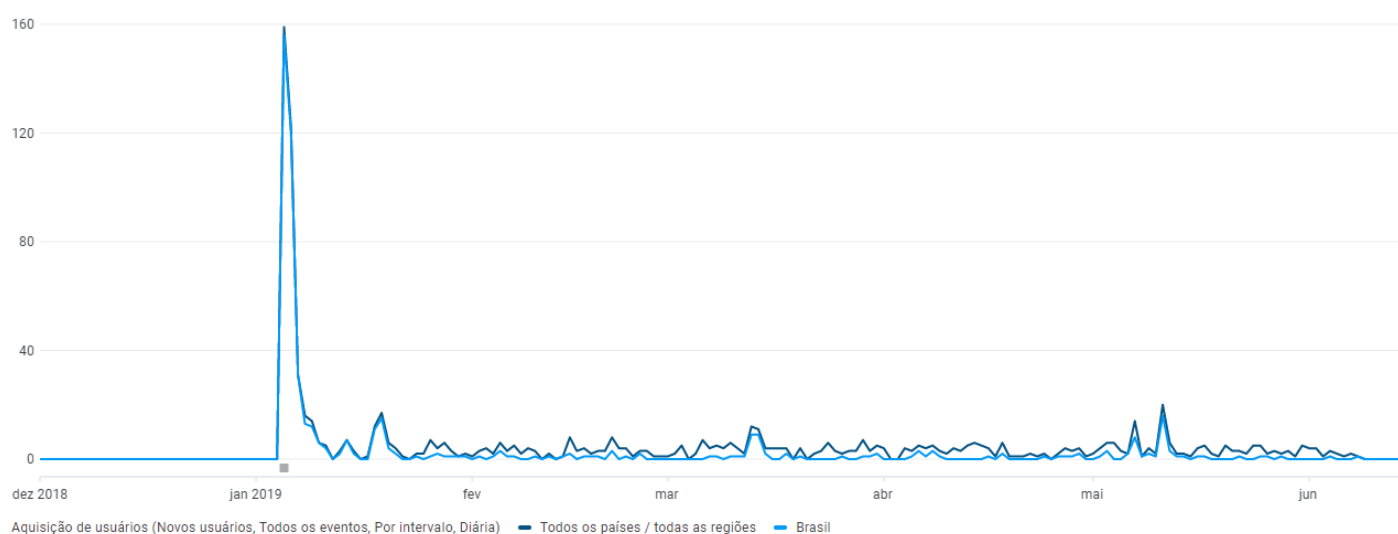


Fig. 1.1: Downloads do aplicativo distribuídos no tempo.

Em julho de 2019 o aplicativo foi retirado da Play Store devido à ausência de uma política de privacidade. Isso deu início a um longo hiato no desenvolvimento que só foi retomado no início de 2020.

A estratégia de monetização usada foi a de mostrar propaganda a partir dos serviço AdMob da Google. No tempo que o aplicativo ficou disponível para download (5 meses), somente com publicidade exibida durante as partidas, foi possível angariar 14 dólares. Não foi possível resgatar o valor, uma vez que só se pode fazer isso em pacotes de 100 dólares.

Considerando que o aplicativo só ficou disponível por 5 meses e que estava em seu estado base, sem qualquer atualização ou melhoramento, considera-se este retorno um ótimo resultado. Espera-se ultrapassar essa marca com o novo lançamento em 2021.

1.1 Sobre o papel do desenvolvimento de jogos na equipe

Este é um assunto de grande complexidade.

Na ciência da computação, o assunto é tratado como um dos mais difíceis, uma vez que, assim como na criação de foguetes, é uma prática extremamente multidisciplinar. Dentre as competências que se espera de uma equipe que se propõe a tal papel temos todo tipo de habilidade que oscila entre técnico e artístico de forma dramática.

Tendo a diversidade do assunto em vista, observa-se uma oportunidade de uma rica interlocução de agentes das mais diversas áreas, assim como vemos de um ponto de vista geral, dentro da equipe de propulsão e tecnologia aeroespacial. Desse forma espera-se que a subárea de desenvolvimento de jogos usufrua dessa diversidade que existe dentro da equipe, e até a expanda ainda mais.

Outro aspecto importante desta prática é a forma como se enxerga a programação. Na pesquisa e na engenharia também é constante a prática de programar, mas isso é feito de forma pragmática e apressada. Esse é o natural, uma vez que o código não é o produto final, mas um meio que se precisa percorrer, seja para uma pesquisa ou para desenvolver uma estrutura ou peça para o foguete. Tudo isso é familiar aos outros integrantes da equipe e para qualquer um da área de engenharia. Mas, dentro da EPTA Entertainment a perspectiva é outra. O código que está sendo desenvolvido é o produto final, e como tal, deve ser feito de forma bem diferente do código acadêmico. Além de funcionar ele deve ser legível e obedecer a padrões de projeto. Além disso, no lugar de exercer uma funcionalidade única e específica, códigos como os desenvolvidos na subárea muitas vezes precisam ser flexíveis e robustos de forma a trabalharem harmonicamente no ecossistema lógico do aplicativo.

Desenvolver essa outra perspectiva de desenvolvimento de software é extremamente positivo a qualquer engenheiro, uma vez que mais e mais o engenheiro moderno se vê cercado por programação, em qualquer especialidade.

1.2 O Mínimo produto viável(MVP)

O jogo para celular "Epta space program" é um programa desenvolvido para Android cujo público alvo são os entusiastas da indústria aeroespacial. Desse forma, o objetivo da equipe é criar algo acessível e informativo.

Capítulo 2

Estado do aplicativo

O aplicativo já se encontra em um estado avançado de desenvolvimento, consequência dos anos de trabalho que se foram. Nesse capítulo será descrito com ele se encontra no início de 2021.

Dentre as coisas descritas temos os seguintes tópicos:

- Disposição dos arquivos.
- Disposição de Game Objects.
- Ecossistema lógico.

2.1 Disposição de arquivos

O gerenciamento de arquivos é muito importante no desenvolvimento de jogos.

O objetivo é organizar a informação, uma vez que existem muitos arquivos (centenas). E podem ser figuras, música, scripts, meta dados, e objetos lógicos de jogo, como prefabs, animações, cenas e muitas outras coisas. Dessa forma é muito importante que tais arquivos sejam dispostos de uma forma inteligível, uma vez que mexer neles faz parte do cotidiano dos desenvolvedores.

Existem duas pastas importantes que devemos saber a composição. A pasta Assets e a pasta Adm. Ambas estão localizadas na pasta principal do projeto Unity do aplicativo.

Na primeira se coloca tudo que entra no desenvolvimento do jogo. Entre as coisas que estão lá dentro temos os scripts, músicas, prefabs, etc... Toda vez que se quer modificar algo no jogo, um dos arquivos dentro desta pasta será modificado.

Na pasta adm são colocados arquivos que não são usados no desenvolvimento do jogo, mas ainda sim são importantes para esse desenvolvimento. Entre as coisas presentes nessa pasta temos o relatório que documenta a criação do aplicativo (como este que estão lendo), a política de privacidade que usamos no lançamento do aplicativo, a chave de publicação, e outros detalhes.

2.2 Disposição de Game Objects

2.3 Ecossistema lógico


```
1      using UnityEngine;
2
3      public class Destroy_obstacles : MonoBehaviour
4      {
5
6          // Destroi tudo que encostar neste objeto
7          void OnCollisionEnter2D(Collision2D coll)
8          {
9              Destroy(coll.gameObject);
10         }
11     }
```

Fig. 2.1: Código do script "Destroy_obstacles.cs".

```
1      using UnityEngine;
2
3      public class audio_controls: MonoBehaviour
4      {
5          // Reference to the scene audio controll
6          private GameObject music_volume_control;
7
8          // makes the starting volume change in UI
9          [SerializeField]
10         private float volume;
11
12         // Start is called before the first frame update
13         void Start()
14         {
15             // Acha o game audio principal, com a música do jogo via tag (
16             //   GameObject Audio Source)
17             music_volume_control = GameObject.FindWithTag("audio_control");
18
19             // Declara o volume
20             music_volume_control.GetComponent<AudioSource>().volume =
21                 volume;
22
23         }
24
25         // Quando o valor do volume muda, atualiza o valor no Audio source
26         // da música
27         public void volume_controll(float volume){
28             music_volume_control.GetComponent<AudioSource>().volume =
29                 volume;
30         }
31     }
```

Fig. 2.2: Código do script "audio_controls.cs".

Capítulo 3

Plano de gestão para 2021

Capítulo 4

Planos futuros