

UNIVERSIDADE DE SÃO PAULO

Informática

**Laboratório**

**Introdução a Ciência da Computação**

Disciplina: SSCO502

Professor: Seiji Isotani

Turma: Info12

Nome: Eduardo Sigrist Ciciliato número USP:7986542

Nome: Willian Fagner de Souza Policiano número USP:8066190

**OBJETIVO**

O projeto 4Physics foi desenvolvido com o intuito de aprendizagem.

**RESUMO**

O projeto 4Physics é um jogo para computadores desenvolvido por Willian e Eduardo como parte fundamental das aulas de Laboratórios de Introdução a Ciência da Computação, ministrada pelo professor Seiji Isotani.

O jogo baseia-se no estilo plataforma no qual o jogador deve atravessar a fase manipulando a gravidade.

Muitos desafios foram mostrados, porém todos foram solucionados direta ou indiretamente.

**ÍNDICE**

1. **INTRODUÇÃO..................................................................................................................1**
2. **MÉTODOS E FERRAMENTAS......................................................................................1**
   1. **Ferramentas.......................................................................................................................1**
      1. **Code::Blocks 10,05.......................................................................................................1**
      2. **SDL................................................................................................................................1**
      3. **Fireworks e Flash..........................................................................................................2**
   2. **Métodos..............................................................................................................................2**
      1. **Classes............................................................................................................................2**
      2. **Headers..........................................................................................................................3**
      3. **Metodos e Lógica aprendidos em aula........................................................................3**
3. **DESAFIOS..........................................................................................................................3**
4. **RESULTADOS E DISCUÇÃO.........................................................................................4**
5. **CONCLUSÃO.....................................................................................................................4**
6. **REFERÊNCIAS..................................................................................................................4**

**1.INTRODUÇÃO**

Como trabalho pratico da disciplina ICC-Lab os alunos deverim escolher um tipo de software para desenvolver. Não houve restrição. O projeto poderia ser desde pequenos programas de gerenciamento de estoque, até grandes jogos e sistemas bancários.

O grupo formado por Willian, Eduardo e Vinícios escolheu a opção de criar um jogo. Este jogo se baseia no tema Sci-Fi e ainda não possui um enrredo. Seu objetivo é atravessar a fase até o seu final usando manipulação de gravidade que é trocada através de blocos especiais do cenário.

Seu público alvo são os Geeks, crianças e todos que gostam de perder um pequeno tempo se divertindo com jogos.

**2. MÉTODOS E FERRAMENTAS USADAS**

Para o desenvolvimento do jogo foi utilizado varias ferramentas e métodos, detre eles, foi usado a biblioteca gráfica SDL, o IDE Code::Blocks, classes, headers e técnicas de programação e lógica vistos em aula. Para renderização e criação das imagens foram utilizados os programas da Adobe, FireWorks e Flash.

**2.1 Ferramentas:**

**2.1.1. Code::Blocks 10.05**

Code::Blocks (ou C::B) é um [ambiente de desenvolvimento integrado](http://pt.wikipedia.org/wiki/Ambiente_de_desenvolvimento_integrado) de [código aberto](http://pt.wikipedia.org/wiki/C%C3%B3digo_aberto) e [multiplataforma](http://pt.wikipedia.org/wiki/Multiplataforma). Ele está sendo desenvolvido em [C++](http://pt.wikipedia.org/wiki/C%2B%2B), usando [wxWidgets](http://pt.wikipedia.org/wiki/WxWidgets). Sua arquitetura é orientada a plugin, de forma que suas funcionalidades são definidas pelos plugins fornecidos a ele. Atualmente, Code::Blocks é voltado para o desenvolvimento em [C](http://pt.wikipedia.org/wiki/C_(linguagem_de_programa%C3%A7%C3%A3o))/C++.

O Code::Blocks está sendo desenvolvido sob [Windows](http://pt.wikipedia.org/wiki/Microsoft_Windows) e [Linux](http://pt.wikipedia.org/wiki/Linux). Alguns usuários compilaram o Code::Blocks com sucesso para [FreeBSD](http://pt.wikipedia.org/wiki/FreeBSD) e [Mac OS X](http://pt.wikipedia.org/wiki/Mac_OS_X) (com alguns problemas na interface).

Este programa não lançava uma versão estável, desde [2008](http://pt.wikipedia.org/wiki/2008). No lugar lançava as chamadas [Nightly builds](http://forums.codeblocks.org/index.php?board=20.0). Entretanto, foi lançada uma nova versão estável (10.05) em [30 de Maio](http://pt.wikipedia.org/wiki/30_de_Maio) de [2010](http://pt.wikipedia.org/wiki/2010).

**2.1.2. SDL**

Simple DirectMedia Layer (SDL) é uma [biblioteca](http://pt.wikipedia.org/wiki/Biblioteca_de_rotinas) multimídia e multiplataforma escrita em [C](http://pt.wikipedia.org/wiki/Linguagem_C) (mas diretamente compatível com C++ e possui interfaces para outras linguagens de programação, como [Ada](http://pt.wikipedia.org/wiki/Ada_(linguagem_de_programa%C3%A7%C3%A3o)), [Eiffel](http://pt.wikipedia.org/wiki/Eiffel), [Java](http://pt.wikipedia.org/wiki/Java), [Lua](http://pt.wikipedia.org/wiki/Lua_(linguagem_de_programa%C3%A7%C3%A3o)), [ML](http://pt.wikipedia.org/wiki/ML), [Perl](http://pt.wikipedia.org/wiki/Perl), [PHP](http://pt.wikipedia.org/wiki/PHP), [Pike](http://pt.wikipedia.org/wiki/Pike), [Python](http://pt.wikipedia.org/wiki/Python), e [Ruby](http://pt.wikipedia.org/wiki/Ruby_(linguagem_de_programa%C3%A7%C3%A3o))), que cria uma abstração em várias plataformas de [gráficos](http://pt.wikipedia.org/w/index.php?title=Gr%C3%A1ficos_de_computador&action=edit&redlink=1), sons, e entrada de [APIs](http://pt.wikipedia.org/wiki/Application_Programming_Interface), tornando possível ao programador escrever um [jogo de computador](http://pt.wikipedia.org/wiki/Jogo_de_computador) ou outra aplicação [multimedia](http://pt.wikipedia.org/wiki/Multimedia) já que ela pode rodar em [GNU/Linux](http://pt.wikipedia.org/wiki/GNU/Linux), [Windows](http://pt.wikipedia.org/wiki/Microsoft_Windows), [Mac OS Classic](http://pt.wikipedia.org/wiki/Mac_OS), [Mac OS X](http://pt.wikipedia.org/wiki/Mac_OS_X), [BeOS](http://pt.wikipedia.org/wiki/BeOS), [FreeBSD](http://pt.wikipedia.org/wiki/FreeBSD), [OpenBSD](http://pt.wikipedia.org/wiki/OpenBSD),[Solaris](http://pt.wikipedia.org/wiki/Solaris_Operating_Environment), [QNX](http://pt.wikipedia.org/wiki/QNX), [IRIX](http://pt.wikipedia.org/wiki/IRIX), e muitas outras plataformas não oficialmente suportadas. Ela gerencia [video](http://pt.wikipedia.org/wiki/Video), eventos, audio digital, CD-ROM, som, threads, processamento de objetos compartilhados, rede e tempo.

**5.1.3 Fireworks e Flash**

O Fireworks é um [editor de imagens](http://pt.wikipedia.org/wiki/Editor_de_imagens) desenvolvido pela [Macromedia](http://pt.wikipedia.org/wiki/Macromedia), posteriormente adquirido pela [Adobe](http://pt.wikipedia.org/wiki/Adobe). Suas funcionalidades focam a publicação gráfica na Internet, por isso inclui suporte a [GIF](http://pt.wikipedia.org/wiki/GIF) animado, [PNG](http://pt.wikipedia.org/wiki/PNG) e imagens fatiadas, além de possuir ótima [compressão de imagens](http://pt.wikipedia.org/wiki/Compress%C3%A3o_de_imagens). A partir da versão MX, ganhou integração com outros produtos da mesma linha,[Dreamweaver](http://pt.wikipedia.org/wiki/Dreamweaver), [Flash](http://pt.wikipedia.org/wiki/Flash) e [Freehand](http://pt.wikipedia.org/wiki/Freehand).

Com a aquisição da [Macromedia](http://pt.wikipedia.org/wiki/Macromedia) em [2005](http://pt.wikipedia.org/wiki/2005) pela [Adobe](http://pt.wikipedia.org/wiki/Adobe), houve um choque entre os seus programas, Fireworks e [ImageReady](http://pt.wikipedia.org/w/index.php?title=ImageReady&action=edit&redlink=1), [Dreamweaver](http://pt.wikipedia.org/wiki/Dreamweaver) e [GoLive](http://pt.wikipedia.org/wiki/GoLive), [Freehand](http://pt.wikipedia.org/wiki/Freehand) e [Illustrator](http://pt.wikipedia.org/wiki/Illustrator). Assim, o objetivo da [Adobe](http://pt.wikipedia.org/wiki/Adobe) foi em criar um conjunto mais poderoso de soluções para a criação, gerenciamento e entrega de conteúdo e experiências atraentes em múltiplos sistemas operacionais, dispositivos e mídias.

O desenvolvimento do ImageReady foi cancelado prevalecendo o Fireworks como editor gráfico para Internet.

Adobe Flash (antes: Macromedia Flash), ou simplesmente Flash, é um software primariamente de [gráfico vetorial](http://pt.wikipedia.org/wiki/Gr%C3%A1fico_vetorial) - apesar de suportar imagens [bitmap](http://pt.wikipedia.org/wiki/Bitmap) e [vídeos](http://pt.wikipedia.org/wiki/V%C3%ADdeo) - utilizado geralmente para a criação de [animações](http://pt.wikipedia.org/wiki/Anima%C3%A7%C3%A3o) interativas que funcionam embutidas num [navegador](http://pt.wikipedia.org/wiki/Navegador) web e também por meio de desktops, celulares, smartphones, tablets e televisores. O produto era desenvolvido e comercializado pela [Macromedia](http://pt.wikipedia.org/wiki/Macromedia), empresa especializada em desenvolver [programas](http://pt.wikipedia.org/wiki/Programa) que auxiliam o processo de criação de páginas web.

**2.2 Métodos:**

**2.2.1 Classes**

Existem duas categorias de tipos de dados usuais em C++, são classificados como tipos básicos e tipos definidos pelo programador.

Assim como na linguagem C, podemos definir dados compostos por associações dos tipos básicos, estes tipos são chamados de estruturas (structs). C++ traz uma nova representação de dados, muito semelhante na forma às estruturas, porém diferentes na forma conceitual: a palavra chave class, que é usada para criar uma classe de objetos mais rica que as structs. Ao declararmos um identificador, tal qual fazemos ao declarar uma variável e no lugar do tipo especifiquemos uma classe criaremos um objeto.

**2.2.2 Headers (cabeçalhos)**

Arquivos-cabeçalhos são aqueles que temos mandado o compilador incluir no início de nossos exemplos e que sempre terminam em .h. A extensão .h vem de header (cabeçalho em inglês). Já vimos exemplos como stdio.h, conio.h, string.h. Estes arquivos, na verdade, não possuem os códigos completos das funções. Eles só contêm protótipos de funções. É o que basta. O compilador lê estes protótipos e, baseado nas informações lá contidas, gera o código correto. O corpo das funções cujos protótipos estão no arquivo-cabeçalho, no caso das funções do próprio C, já estão compiladas e normalmente são incluídas no programa no instante da "linkagem". Este é o instante em que todas as referências a funções cujos códigos não estão nos nossos arquivos fontes são resolvidas, buscando este código nos arquivos de bibliotecas.

**2.2.3. Metodos e Lógica aprendidos em aula**

Os metodos aprendidos em aula são métodos de programação básica em C tais como o comando if e for, matrizes e vetores, funções, arquivos entre outros

**3. DESAFIOS**

Durante todo o desenvolvimento do software foi encontrado várias dificuldades. Foi necessário aprender a manipular a biblioteca gráfica e um dos integrantes do grupo não conhecia o uso de classes em C++, além disso, houve a necessidade uma queda no número de colaboradores desenvolvedores do software.

Vinícios, o terceiro integrante do nosso grupo, por motivos pessoais, desistiu do curso. Este impasse foi motivo do surgimento da maioria das dificuldades que deparamos.

Criamos uma grande expectativa sobre o projeto 4Physics e por isso nossa carga aumentou. Devido o desligamento do nosso integrante, as atividades ficou dividida de uma forma muito densa entre o restante dos integrantes deste grupo, isto fez com que tivessemos nosso tempo diminuido praticamente 1/3, automaticamente procuramos métodos mais rápidos e práticos para desenvolver este jogo e cumprir o prometido, um destes métodos foi usar classes em C.

Willian ficou encarregado da criação das imagens e das Sprite Sheets e da preparação do ambiente de desenvolvimento em SDL, Eduardo ficou com a maioria da lógica e programação do game. Isso gera também mais uma dificuldade, a de entender a lógica do parceiro de grupo, na qual também foi solucionada.

Além de tudo, como foi preciso aprender novos ambientes, obtivemos vários bugs e problemas de lógica que precisamos corrigir.

Pode se considerar também a dificuldade para pesquisa de tutoriais SDL. O monitor voluntário de ICC, Bruno, contribuiu com a indicação do site Lazy Foo que foi o grande auxiliador deste projeto.

**4. RESULTADOS E DISCUÇÃO**

Os resultados foram melhores do que o esperado e o projeto 4Physics excedeu as expectátivas de visual e desempenho. Do início ao término do projeto usamos aproximadamente 20 horas de desenvolvimento. Como a especifição do projeto não estava expresso a linguagem de programação a ser usada, o 4Physics foi desenvolvido em C++ que é próximo de C e para facilitar a resolução do nosso problema de tempo curto.

**5. CONCLUSÃO**

O projeto 4Physics foi desenvolvido com o intuito de aprendizagem. Realmente proporcionou tanto aprendizado técnico como aprendizado profissional. Foi preciso aprender a gerenciar o tempo, recursos a serem usados, mão de obra, prazos e principalmente superar vários desafios. Durante o projeto, todos estavam envolvidos na busca da excêlencia. Embora ainda não é um programa perfeito e livre de bugs, seu desenvolvimento foi satisfatório. Será dada continuidade no projeto após apresenta;cão em classe.

**6. REFERÊNCIAS**

<http://pt.wikipedia.org/wiki/Code::Blocks>

<http://pt.wikipedia.org/wiki/SDL_(biblioteca)>

<http://pt.wikipedia.org/wiki/Adobe_Flash>

<http://pt.wikipedia.org/wiki/Adobe_Fireworks>

<http://pt.wikibooks.org/wiki/Programar_em_C%2B%2B/Classes>

<http://mtm.ufsc.br/~azeredo/cursoC/aulas/c750.html>

<http://www.lazyfoo.net/SDL_tutorials/>