UTILIZAÇÃO DO ANDROID PARA O DESENVOLVIMENTO DO JOGO *STUDENT COMBAT*

R. S. Silva[1] e B. G. Araújo[2]
E-mails: raul-macintosh@bol.com.br[1]; bruno.gomes@ifrn.edu.br[2]

RESUMO

Todo desenvolvedor de jogos ou aplicativos, busca desenvolver para as plataformas que mais atraem usuários. Mas nem sempre o jogo ou aplicativo é fácil de ser criado. Entretanto, o *Android* (Sistema Operacional da *Google*) é, atualmente, a melhor plataforma para os desenvolvedores, pois vem constantemente atraindo vários usuários, além de ser fácil e eficiente para programar. Com isso, o presente trabalho visa expor um exemplo dessa facilidade e eficiência da plataforma *Android* no desenvolvimento do jogo *StudentCombat*, um jogo de perguntas e respostas para ser usado por estudantes.

PALAVRAS-CHAVE: Android, Dispositivos Móveis, Programação para Android, Jogos Educativos.

THE USE OF ANDROID FOR THE DEVELOPMENT OF THE GAME STUDENT COMBAT

ABSTRACT

Every game developer or applications are looking to develop for the platforms that attract users. But not always the game or application is easy to be created. However, the Android (OS from Google) is currently the best platform for the developers, as has been constantly attracting many users, and it is easy and efficient to program. Aiming at this, this paper aims to expose an example of the ease and efficiency of the Android platform in the development of the Student Combat game, a game of questions and answers to be used by students.

KEYWORDS: Android, Mobile Devices, Development for Android, Educational Games.

1 INTRODUÇÃO

Com a chegada dos *Smartphones* e *Tablets*, a computação móvel cresceu muito nos últimos anos. A demanda por esses aparelhos subiu mais ainda com a criação do Sistema Operacional *Android* em 2008, já que os dispositivos compatíveis eram mais baratos, além de serem livres para o desenvolvimento de aplicativos (ANDROID, 2013). A plataforma disponibiliza ainda todas as ferramentas gratuitas para a criação de qualquer aplicativo.

Junto aos dispositivos móveis, há uma crescente demanda por uma nova metodologia de auxílio no ensino, e os Jogos Educacionais é um dos métodos que mais despertam a atenção dos alunos (VITTA *et al.*, 2012) (ARANHA, 2006) (OLIVEIRA *et al.*, 2009). Esses jogos podem ser usados nos dispositivos móveis, aparelhos de menor poder e processamento que permitem mobilidade e que estão ficando cada vez mais presentes na vida das pessoas (COSTA et al., 2010).

Devido a isso, o presente trabalho propõe o desenvolvimento de um jogo para a plataforma *Android*, no intuito de auxiliar estudantes do ensino médio em todas as disciplinas de forma divertida e estimulante.

O jogo é chamado de *Student Combat*, e consiste em uma disputa entre dois estudantes utilizando diferentes dispositivos conectados à internet, onde ambos irão responder questões referentes a uma determinada disciplina. O jogo finaliza no momento que um dos estudantes erra uma das perguntas dando a vitória para o outro, que por sua vez irá aumentar sua pontuação do jogo. Ao final, a sua classificação é atualizada no *Ranking* do *Student Combat*. Também será explicado como foi feito o desenvolvimento do *Student Combat*, quais as ferramentas utilizadas, ambientes de desenvolvimento e trechos de códigos.

2 REVISÃO BIBLIOGRÁFICA

Com o crescimento da tecnologia móvel, as empresas de informática vêm investindo cada vez mais na mobilidade de dispositivos, e a Google é uma delas, com o lançamento do Android em 2008 (ANDROID, 2012). O Android consiste em um Sistema Operacional feito para dispositivos móveis como *Tablets* e *Smartphones* e roda sobre o Kernel Linux (VAUGHAN-NICHOLS, 2012). Ele permite o desenvolvimento de aplicativos na Linguagem de Programação Java, controlando o dispositivo via bibliotecas desenvolvidas pela Google (SHANKLAND, 2012).

Ele é um dos Sistemas Operacionais que mais cresce, se tornando em uma plataforma tão popular quanto Windows, Mac OS e Linux. É possível visualizar o crescimento do número de ativações de dispositivos *Android* de 2009 até 2012 na Figura 1.

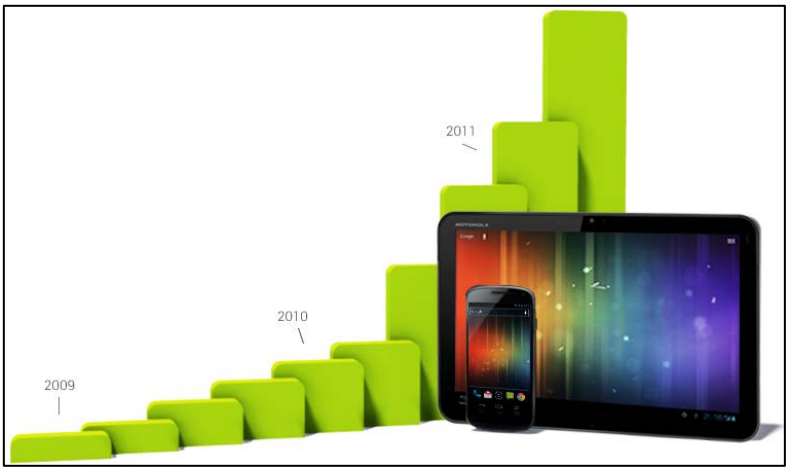

Figura 1: Ativações de dispositivos Android. (ANROID, 2013).

O grande forte do *Android* é o consumo de conteúdo: vídeos, música, livros, navegação na internet, jogos e outras mídias suportadas. Os usuários realizam mais de 1,5 bilhões de downloads no Google Play a cada mês (ANDROID, 2013).

Em relação ao desenvolvimento, os desenvolvedores *Android* possuem a mesma opinião a respeito do Sistema Operacional: A API (*Application Programming Interface* – Interface de programação de aplicativos) é inteligente, bem diversificada e possui uma estrutura bastante simples de compreender. Isso é também o pensamento de empresas que desenvolvem aplicativos e jogos para o *Android*. (OGLIARI, 2012).

3 METODOLOGIA E MÉTODOS

Para o desenvolvimento do jogo foi utilizado a IDE Eclipse, por ser a mais apropriada para o desenvolvimento *Android* e por ter uma boa compatibilidade. Foram também utilizados os kits de desenvolvimento SDK (*Software Development Kit*–Kit de Desenvolvimento de Software), o JDK (*Java Development Kit* – Kit de desenvolvimento Java) e suas APIs. O código do jogo foi escrito na linguagem de programação Java e a interface gráfica em XML. O *Play Framework* foi utilizado na construção do *Web Service* para realizar a comunicação do jogo com o servidor.

O jogo *Student Combat* consiste em conectar dois usuários a um servidor e mostrar para ambos uma mesma questão sobre uma determinada disciplina. Cada questão é composta por um enunciado, quatro alternativas e uma única resposta certa.

As questões, as alternativas e a resposta são armazenadas, em um banco de dados, por professores das disciplinas e são selecionadas aleatoriamente para o jogador. O acesso à questão é feito pelo código demonstrado na Figura 2.

```java
public class HttpCadastro {

    private String q;

    void cadastra(String URL){
    BufferedReader in = null;
     try{
         HttpClient client = new DefaultHttpClient();
           HttpGet request = new HttpGet();
           request.setURI(new URI(URL));
           HttpResponse response = client.execute(request);
           in = new BufferedReader(new InputStreamReader(response.getEntity().getContent()));
           StringBuffer sb = new StringBuffer("");
           String line = "";
           String NL = System.getProperty("line.separator");

           while ((line = in.readLine()) != null) {
               sb.append(line + NL);
           }
           in.close();
           String page = sb.toString();
           System.out.println(page);
           q = page;
     } catch (URISyntaxException e) {
        e.printStackTrace();
    } catch (ClientProtocolException e) {
        e.printStackTrace();
    } catch (IOException e) {
        e.printStackTrace();
    } finally {
           if (in != null) {
               try {
                   in.close();
                   } catch (IOException e) {
                   e.printStackTrace();
               }
      }
     }
    }

    String getInformacao(){
        return q.trim();
    }
}
```

Figura 2: Método HTTP usado para puxar uma questão do banco de dados.

O método HTTP foi utilizado para que através de uma URL o aplicativo armazene em uma variável (declaração da variável na linha 17 e armazenagem do texto na linha 37) o enunciado da questão juntamente com cada alternativa e a resposta correta, todos separados por uma barra vertical "|". Esse método está separado em uma classe distinta para poder ser usado tanto para buscar as questões, quanto para cadastrar ou buscar um jogador, uma vez que basta modificar a URL.

Após o armazenamento da questão, é necessário que se separe o enunciado, as alternativas e as respostas. Para isso a classe responsável por expor a questão na tela é dotada de um código específico para a quebra da variável que contém o enunciado, expor o enunciado e as alternativas e guardar a resposta certa em outra variável para ser comparada com a alternativa marcada pelo jogador.

Na Figura 3 é demonstrada a quebra e a exposição do enunciado com as alternativas.

```
40          HttpCadastro q = new HttpCadastro();
41          q.cadastra(""); // URL da disciplina.
42
43          questao = (TextView) findViewById(R.TextView.Questao);
44          alternativa1 = (RadioButton) findViewById(R.RadioButton.Alternativa1);
45          alternativa2 = (RadioButton) findViewById(R.RadioButton.Alternativa2);
46          alternativa3 = (RadioButton) findViewById(R.RadioButton.Alternativa3);
47          alternativa4 = (RadioButton) findViewById(R.RadioButton.Alternativa4);
48
49          StringTokenizer var = new StringTokenizer(q.getInformacao, "|");
50          questao.setText(var.nextToken());
51          alternativa1.setText(var.nextToken());
52          alternativa2.setText(var.nextToken());
53          alternativa3.setText(var.nextToken());
54          alternativa4.setText(var.nextToken());
55          resposta = var.nextToken();
56
57          numero = Integer.parseInt(resposta.trim());
```

Figura 3: Classe StringTokenizer sendo usada para a segregação de uma variável.

A classe StringTokenizer (presente no JDK 1.0) separa a variável sempre que há uma barra vertical. Cada pedaço separado é mostrado na tela do usuário (na linha 50 o enunciado é mostrado, da linha 51 a 54 são mostrados as alternativas da 1ª a 4ª e na linha 55 o número da alternativa correta é armazenado na String "resposta"), com exceção da resposta. Na linha 57 a String "resposta" é convertida para um valor do tipo inteiro e armazenada na variável "numero".

Na Figura 4, é possível visualizar a verificação da resposta correta.

```
59          RadioGroup rg = (RadioGroup) findViewById(R.RadioGroup.Alternativas);
60          rg.setOnCheckedChangeListener(new RadioGroup.OnCheckedChangeListener() {
61              public void onCheckedChanged(RadioGroup group, int checkedId) {
62                  boolean resposta1 = R.RadioButton.Alternativa1 == checkedId;
63                  boolean resposta2 = R.RadioButton.Alternativa2 == checkedId;
64                  boolean resposta3 = R.RadioButton.Alternativa3 == checkedId;
65                  boolean resposta4 = R.RadioButton.Alternativa4 == checkedId;
66                  if (resposta1){
67                      verificacao = 1;
68                  }else if (resposta2){
69                      verificacao = 2;
70                  }else if (resposta3){
71                      verificacao = 3;
72                  }else if (resposta4){
73                      verificacao = 4;
74                  }
75              }
76          });
77      }
```

Figura 4: Verificação da alternativa marcada.

Da linha 66 a linha 74, na Figura 4, ocorre a inserção de um número na variável "verificação" que posteriormente irá comparar com o valor da variável "numero" (Figura 3) e dependendo dessa comparação o jogador será transferido para uma tela informando sua vitória ou derrota.

Caso ocorra um empate entre os jogadores, ambos irão continuar a responder as questões até que um deles erre e dê a vitória ao seu oponente.

4 RESULTADOS E DISCUSSÕES

O jogo *Student Combat* foi desenvolvido utilizando a tecnologia Java para a plataforma *Android*. Sua tela inicial é demonstrada na Figura 5. Ela é composta por dois botões, o de Login e o de Cadastrar. O botão Login irá transferir o jogador para uma tela onde ele informará seu nome de usuário e sua senha. O botão de Cadastrar irá transferir o jogador para uma tela onde ele fará o seu cadastro no jogo, caso não tenha uma conta já realizada.

Figura 5: Tela inicial do *StudentCombat*.

Depois de ter sido feito o *login* do usuário, o jogo mostrará os jogadores que estão online, onde o usuário escolherá um para desafiar, ativando o respectivo botão de rádio e em seguida clicando em "Desafiar", como mostra a Figura 6.

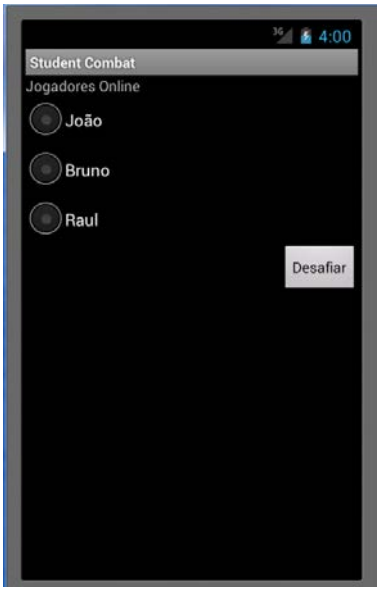

Figura 6: Tela de jogadores online.

Após clicar em“Desafiar”, e aguardado o tempo de aceitação do oponente, a disputa é iniciada. Uma tela com um enunciado, quatro alternativas e um botão “Responder”, é mostrada aos dois jogadores (Figura 7), onde só poderá ser marcada uma alternativa e em seguida o botão “Responder” deverá ser clicado.

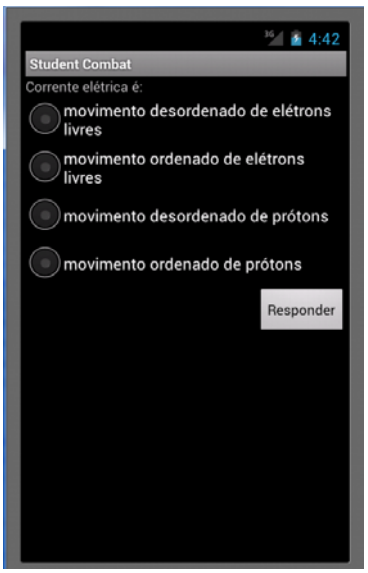

Figura 7: Exemplo de questão em um desafio.

Com a alternativa selecionada e o botão “Responder” pressionado, o jogo irá fazer a comparação da alternativa com a resposta certa e em seguida, caso não haja empate entre os jogadores, cada jogador será levado a uma das telas mostradas na Figura 8, dependendo do resultado das comparações das alternativas.

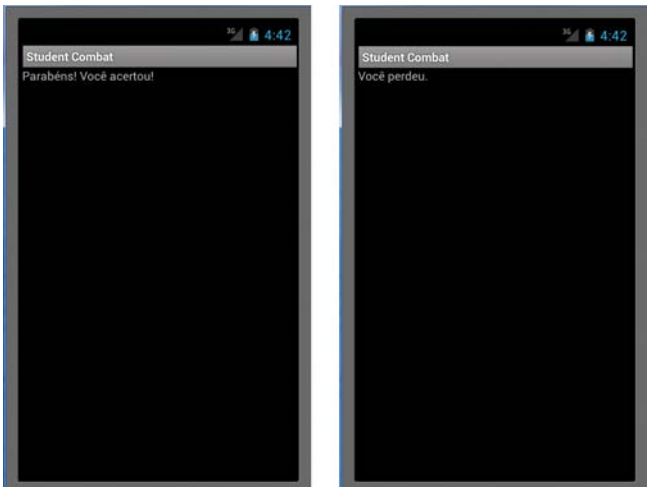

Figura 8: Telas de vitória (esquerda) e derrota (direita).

A união da linguagem de programação Java com o *Android* proporcionou, de forma bastante eficiente e simples, o desenvolvimento do jogo. A disponibilização da classe StringTokenizer na API do Java simplificou a divisão da questão buscada através do método HTTP já preparado para o uso no *Android*.

5 CONCLUSÃO

O desenvolvimento do jogo *Student Combat* foi facilitado devido à disponibilidade de ferramentas gratuitas e eficientes da plataforma *Android*, além da linguagem de programação, Java, ser amplamente difundida pela Internet, de fácil compreensão e também possuir uma API repleta de classes que minimizam o trabalho de um desenvolvedor de aplicativos.

A primeira versão do *Student Combat* foi desenvolvida com o propósito de ser uma alternativa divertida para auxiliar no processo de aprendizagem dos alunos do ensino médio, proporcionando, através de um jogo de perguntas e respostas, uma melhora na fixação dos conteúdos ministrados nas disciplinas escolares.

Como trabalhos futuros, estão sendo realizadas melhorias na interface gráfica, como planos de fundo e animações, para que o jogo seja tão atrativo quanto os que são, comumente, encontrados em dispositivos móveis. Pretende-se também acrescentar outros tipos de desafios e também permitir que o jogador crie salas de jogos com senhas para que o jogo possa ser restrito a algumas pessoas. Após a conclusão da versão final do jogo, pretende-se publicá-lo na *Play Store* para que qualquer um com um dispositivo móvel com *Android* possa baixá-lo.

6 REFERÊNCIAS BIBLIOGRÁFICAS

ANDROID. Android, the world's most popular mobile platform. Disponível em: <http://developer.android.com/about/index.html>. Acesso em: 16 mai 2013.

ARANHA, G. Jogos Eletrônicos como um conceito chave para o desenvolvimento de aplicações imersivas e interativas para o aprendizado. In: Ciências & Cognição. V.07,p. 105-110, 2006.

COSTA, J. H. L.; SILVA, H. C. A.; NASCIMENTO, G. F. C. L. A Questão dos Jogos Eletrônicos para Inclusão Digital e Social no Contexto da Biblioteconomia e Ciência da Informação. Encontro Nacional de Estudantes de Biblioteconomia, Documentação, Gestão e Ciência da Informação, 2010, João Pessoa. Anais do 33º Encontro Nacional de Estudantes de Biblioteconomia, Documentação, Gestão e Ciência da Informação (ENEBD), 2010.

OGLIARI, R. As principais facilidades no desenvolvimento Android. Disponível em: <http://itweb.com.br/blogs/as-principais-facilidades-no-desenvolvimento-android/>. Acesso em: 16 mai 2013.

OLIVEIRA, W. P.; SCHIMIGUEL, J.; SILVEIRA, I. F.; Araújo JR, C. F.; AMARAL, L. H.; OLIVEIRA, I. C. A.; VEIGA, J. S. Desenvolvimento de Simulações Web para uma Atividade de Ensino-Aprendizagem. In: Revista Network Technologies - Faculdades Network. v.3, n.1, 2009.

ORACLE. ClassStringTokenizer. Disponível em: <http://docs.oracle.com/javase/1.4.2/docs/api/java/util/StringTokenizer.html>. Acesso em: 19 mai 2013.

SHANKLAND. Stephen. Google's Android parts ways with Java industry group.Disponível em: <http://news.cnet.com/8301-13580_3-9815495-39.html>. Acesso em: 10 mai 2013.

VAUGHAN-NICHOLS, S. J. Android/Linux kernel fight continues. Disponível em: <http://blogs.computerworld.com/16900/android_linux_kernel_fight_continues>. Acesso em: 10 mai 2013.

VITTA, A.; VITTA, F. C. F.; GATTI, M. A. N.; SIMEÃO, S. F. A. P. Educative Games and Expositive Lesson: Comparison of Educational Techniques on Sitting Posture. In: Journal of Human Growth and Development. V.22, n1, p. 47-52, 2012.