PES
Institute of Technology

V SEMESTER SPECIAL TOPIC

SmartDots

By:
Sadan Sohan M
-1PI09CS076
Sandeep Raju P
-1PI09CS081

Guided By:
Krupesha D
Professor
Computer Science Dept.
PESIT, Bangalore

PES

Institute of Technology

Certificate

This is to certify that **Sadan Sohan M (1PI09CS076)** has satisfactorily completed the Special Topic titled

'SmartDots'

prescribed by PES Institute of Technology (Autonomous Institute under VTU) in the year 2011 (August - December).

Signature of Guide :
Prof. Krupesha D

Signature of HOD :
Prof. Nitin V Pujari

Acknowledgment

We sincerely thank the department for giving us an opportunity to explore new avenues and implement the theoretical knowledge that we have learnt. We wish to express our sincere gratitude to Prof. Nitin V Pujari, HOD, Computer Science Department, PESIT for providing us with an opportunity to carry out the special topic on 'SmartDots'. We also wish to express our gratitude to Prof. Krupesha D, Computer Science Department, PESIT for providing us with the guidance needed. A special mention must be made about the Nokia Mobile Innovation Lab that provided us with all the necessary resources and gave us enriching support.

Abstract

SmartDots is a two player classic board game. The board consists of required number of dots. The game starts with each player connecting any two dots of his choice with a line. Thus each player marks a single line in his/her turn. The primary motive of the game is to complete as many boxes as possible. The player who completes a box in his/her turn gets to play the next chance as well. The game can be played against the phone or against any one you wish. To play with the phone an exclusive AI algorithm has been designed.

The game also boasts of the reverse implementation of the normal game where in the user has to achieve as few boxes as possible to win. It comprises of different levels to enable the user to smoothly sail through the game. It has an interactive user interface design. It involves the storing of level specific high score.

Contents

1 Introduction

SmartDots is a mobile application developed for Qt enabled Nokia mobile phones. It is a strategy game which has a grid of dots. The user should draw a line between the dots by taking turns alternatively. The aim of the game is to make complete boxes by joining dots. The final winner will be the one who has the most number of boxes in the grid. An exclusive algorithm has been developed so a player can play with the phone just like he plays with any other player. A view of the game in progress is shown in the figure.

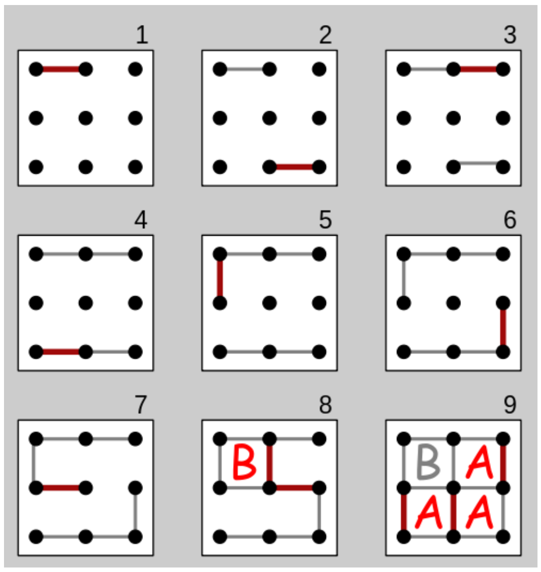


Figure 1: Sample play of the game

2 Problem Definition

The problem definition of this project was simple. We had to implement the classic game of dots and boxes in a touch screen mobile phone with an attractive user interface design. The game also required a robust algorithm which could compete with the player in one player mode of the game. This algorithm was required to be flexible enough to cater to all the levels of the game. We also had to implement a reverse version of the game and thus an algorithm for reverse dots as well had to be designed.

3 Literature Survey

We made extensive research on how the game is played and got ourselves familiar with the game. We went about searching on how to implement the game on such a small screen and with only limited resources.

Wikipedia (http://en.wikipedia.org/wiki/Dots_and_Boxes) provided with the complete rules of the game and all its variations. We chose Qt as we were already familiar with it.

As 'Qt' is an opensource platform there was no dearth for learning resources. The official Qt documentation (http://doc.qt.nokia.com) provided more than enough resources to learn. Nokia developer website (http://www.developer.nokia.com) provided with all the developer tools. One of the major source of resource and help was the #qt-qml IRC (Internet Relay Chat) channel on freenode (www.freenode.net).

The W3C website (http://www.w3schools.com) provides a wealth of resource for the JavaScript programming language.

Tizag tutorials (http://www.tizag.com/javascriptT/) provided us with all the required help for JavaScript.

4 Project Requirement Definition

The development involves the use of a declarative language called QML for User Interface design. The User Interface involves a lot of creativity in designing customized elements for enhanced user experience. Adobe Photoshop CS5 is one of the widely used tools for this purpose. These customized elements had to be integrated with QML to achieve the full User-Interface of the game. The AI for the game was initially developed using C++, for the pure reasons of better familiarity. Later on, it was converted to JavaScript to integrate it with the User Interface. As C++ is statically scoped and JavaScript is dynamically scoped, the conversion was a tedious process.

Two important softwares were required for this game

1. Adobe Photoshop CS5
2. Qt Creator

4.1 Adobe Photoshop CS5

Adobe Photoshop CS5 is a professional image manipulation software user all over. This software provides with various options for graphics designing and rapid development of User Interface.

4.2 Qt Creator

Qt Creator is a standard IDE for application development in Qt. It has all the tools required to build mobile applications. It consists of an inbuilt emulator to debug and test the applications on the fly. The IDE also provides facilities to test and debug our applications remotely via the 'remote compiler option'.

5 System Requirement Specification

The basic requirements of the phone required to run SmartDots is as follows

1. Touchscreen mobile phone having a minimum resolution of 640px $\times$ 480px and have Qt platform installed in them
2. Mobile phones having any of these Operating systems
   (a) Symbian
   (b) Meego
   (c) Maemo
   (d) Windows Phone 7.5
   (e) Android

6 System Design

System design is the bridge between system and requirements analysis and system implementation. Some of the essential fundamental concepts involved in the design of applications are

1. Abstraction
2. Modularity
3. Verification

6.1 Abstraction

Abstraction is used to construct solutions to problems without having to take account of the intricate details of the various components sub-programs. Abstraction allows system designer to make step-wise refinements by which at each stage of the design unnecessary details associated with representation or implementation may be hidden from the surrounding environment.

6.2 Modularity

Modularity is concerned with decomposing of main module into well defined, manageable units with well-defined interfaces among the units. This enhances design, clarity, which in turn eases implementation, debugging, testing and documentation maintaining of the software product.

6.3 Verification

Verification is a fundamental concept in software design. It can be demonstrated that the design will result in an implementation, which satisfies the customers' requirements.

7 Detailed Design

Our application consisted of two parts.

1. UI design using QML
2. Algorithm design using JavaScript

The interaction between UI and Algorithm and its place in Qt application structure is as shown in the figure.

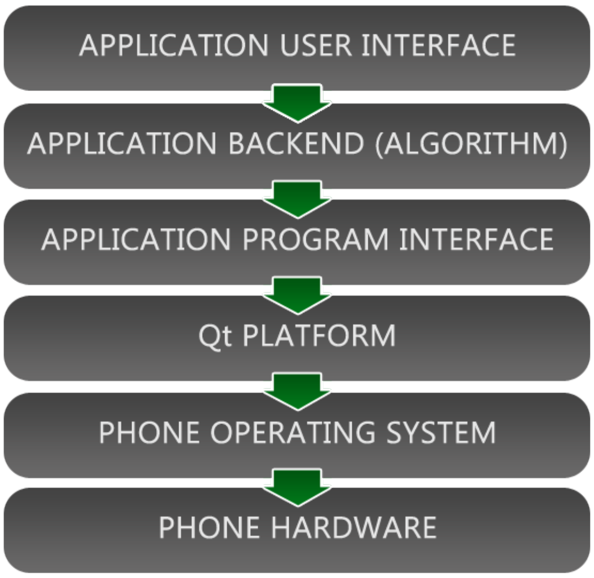

APPLICATION USER INTERFACE
APPLICATION BACKEND (ALGORITHM)
APPLICATION PROGRAM INTERFACE
Qt PLATFORM
PHONE OPERATING SYSTEM
PHONE HARDWARE

Figure 2: Qt application structure and its hierarchy in the system

Some of the main modules in the algorithms are as follows

7.1 AI Coordinate Generator

```
function AIcoordinateGenerator()
{
 if(level is beginner)
   /*In this level artificial intelligence intentionally
   gives up a box for the user in the 24th move of the game*/
    if(it is move no. 24)

        getThree();
        /* getThree() is a method which
        finds a box with 3 lines marked. */
        //mark the appropriate line in the appropriate box.
          if(getThree = null)
            getTwo();
        /*getTwo() is a method which
        finds a box with 2 lines marked*/
        //mark the appropriate line in the box.
             if(getTwo = null)
```

```
              getOne();
          /*getOne() is a method which
         finds a box with 1 lines marked*/
       //mark the appropriate line in the box.
               if(getOne = null)
                  getZero();
         /*getZero() is a method which
         finds a box with 0 lines marked*/
       //mark the appropriate line in the box.

      else

        getThree();

       //mark the appropriate line in the appropriate box.
         if(getThree = null)
         getOne();

       //mark the appropriate line in the box.
           if(getOne = null)
            getZero();

       //mark the appropriate line in the box.
              if(getZero = null)
                getTwo();

       //mark the appropriate line in the box.

else if(level is intermediate)
  /*In this level artificial intelligence
  unintentionally gives up a box for the user*/

    getThree();

       //mark the appropriate line in the appropriate box.
         if(getThree = null)
         getOne();

       //mark the appropriate line in the box.
           if(getOne = null)
            getZero();

       //mark the appropriate line in the box.
              if(getZero = null)
                getTwo();

       //mark the appropriate line in the box.

 else if(level is expert)
  /*In this level artificial
```

```
   intelligence never gives up a box*/

     getThree();

        //mark the appropriate line in the appropriate box.
          if(getThree = null)
          getOne();

        //mark the appropriate line in the box.
            if(getOne = null)
             getZero();

        //mark the appropriate line in the box.
               if(getZero = null)
                 getTwo();

        //mark the appropriate line in the box.
}
//end of AIcoordinateGenerator()
```

7.2 Other Supporting Functions

```
function getOne()
{
 Select a random box with only 1 line marked
 if(level is expert)
   while(the selected box has boxes around it with 2 lines marked)
         Select another box with 1 line marked
}

function getZero()
{
 Select a random box with no lines marked
 if(level is expert)
   while(the selected box has boxes around it with 2 lines marked)
         Select another box with no lines marked

}

function setWinnerFlagIf4(row,col)
{
 //update the database if a box has been completed
 //receives the row and column of the marked line as parameters
 /*tempSum is a variable which keeps track
 of whether a box has been completed. It is set to 4 if a box
 has been completed or else it is set to 0*/
 calculate tempSum
 if(tempSum = 4)
   set winnerFlag to 1
   /*winnerFlag is a variable used to highlight
   the appropriate box in the User Interface.*/
   //no player switch.
 else
   switch the player
}
```

8 Implementation

The game was implemented majorly in two programming languages

1. QML (Qt Meta Language)
2. JavaScript

8.1 QML (Qt Meta Language)

QML (Qt Meta Language or Qt Modeling Language) is a JavaScript-based, declarative language for designing user interface centric applications. It is part of Qt Quick, the UI creation kit developed by Nokia within the Qt framework. QML is mainly used for mobile applications where touch input, fluid animations (60 FPS) and user experience are crucial. QML documents describe an object tree of elements. QML elements shipped with Qt are a sophisticated set of building blocks, graphical (e.g., rectangle, image) and behavioral (e.g., state, transition, animation). These elements can be combined to build components ranging in complexity from simple buttons and sliders, to complete internet-enabled programs. QML elements can be augmented by standard JavaScript both inline and via included .js files. Elements can also be seamlessly integrated and extended by C++ components using the Qt framework.

8.1.1 Role of QML in our game

QML was used extensively to design the User Interface and the animations in it. QML consisted of defining the actions for various components of the design. Its was seamlessly combined with the designs generated using Adobe Photoshop CS5.

We have used QML Listview element for the help file. It creates a nice Flickable interface for easy user interaction. For all the animations in the user interface we have used QML Number Animation element which moves the user interface elements according to the defined animation. The QML MouseArea element has been used to create click the mouse areas to interact with the user.

8.2 JavaScript

JavaScript is a prototype-based scripting language that is dynamic, weakly typed and has first-class functions. It is a multi-paradigm language, supporting object-oriented, imperative, and functional programming styles. JavaScript's use in applications outside Web pages for example in PDF documents, site specific browsers, mobile applications and desktop widgets are significant.

8.2.1 Role of JavaScript in our game

JavaScript was extensively used to define and implement the algorithm initially designed in C++. JavaScript beautifully and seamlessly handled the AI algorithm and its responses which should trigger appropriate actions in the User Interface end.

In JavaScript we have used the standard components and data structures like arrays. Extensive use of logging has been made for debugging and testing purposes. The ′console.log ()′ function has been used for this purpose.

9 Integration

We had to cope with two kinds of integrations.

1. Photoshop - QML
2. QML - JavaScript

9.1 Photoshop - QML

This phase of QML formed the front end of our project. It dealt with using images in various aspects of the user interface in order to beautify the game and help user in all possible ways to get through the initial part of the application with ample ease and comfort.

9.2 QML - JavaScript

This formed the back end and quite easily the most challenging part of the project. Relevant parts of the algorithm had to be employed in different areas of the user interface to ensure the correct working of the game. We faced some constraints regarding the integration such as some parts of JavaScript being disabled while being integrated into QML. So we suitably changed the implementation quite a number of times to ensure the integrated final product works absolutely fine.

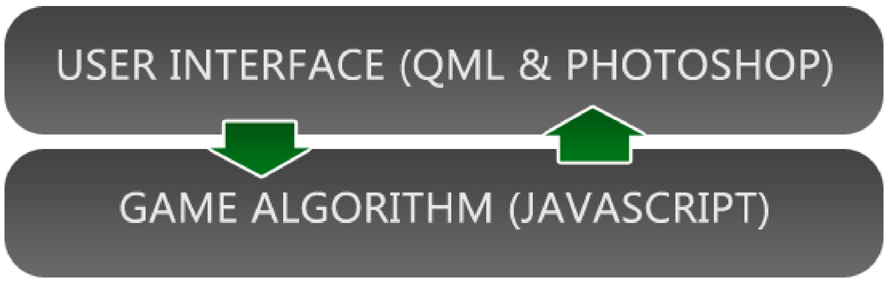


Figure 3: Integration and interfacing of UI and Algorithm

10 Testing

The Fig below shows a five stage testing process where system components are tested, the integrated system is tested and, finally, the system is tested with the customer's data. Ideally, component defects are discovered early in the process and interface problem when the system is integrated. However, as defects are discovered the program must be debugged and this may require other stages in the testing process to be repeated. Errors in program components may come to light during integration testing. The process is therefore an iterative one with information being fed back from later stages to early parts of the process.

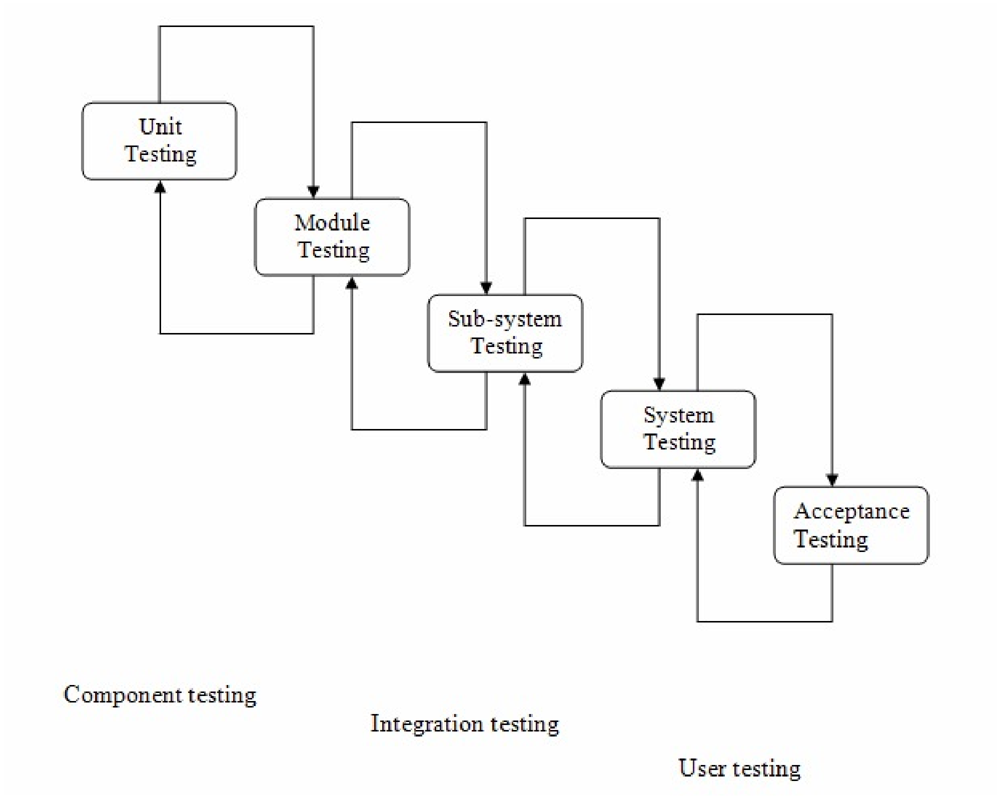


Figure 4: Figure depicting the five stage testing process

10.1 Component Testing

10.1.1 Unit Testing

Unit testing is essential for the verification of the code produced during the coding phase and the goal is to test the internal logic of the module/program. Each and every class of the 'SmartDots' was considered as a single unit and was tested individually to confirm its functional correctness.

10.1.2 Integration Testing

The Integration Testing consists of three parts

1. Module Testing
   A module is a collection of dependent components such as classes and datatypes.

A module encapsulates related components, so can be tested without other system modules.

2. Sub-System Testing
All the tested modules are combined into sub systems, which are then tested. The goal is to see if the modules are properly integrated, and the emphasis being on the testing interfaces between the modules.

3. System Testing
The sub-systems are integrated to make up the system. this process is concerned with finding errors that result from unanticipated interactions between sub-systems and sub-system interface problems. It is mainly used if the software meets its requirements. The reference document for this process is the requirement document.

10.1.3 User Testing

This phase is also called as acceptance testing. This is the final stage in the testing process before the system is accepted for operational use. The system is tested with data supplied by the system customer rather than simulated test data.

10.2 Testing Methods

Testing is a process of executing a program to find out errors. If testing is conducted successfully, it will uncover all the errors in the software. Any testing can be done basing on two ways

10.2.1 White Box Testing

It is a test case design method that uses the control structures of the procedural design to derive test cases. Using this testing a software Engineer can derive the following test cases

1. Exercise all the logical decisions on either true or false sides.
2. Execute all loops at their boundaries and within their operational boundaries.
3. Exercise the internal data structures to assure their validity.

10.2.2 Black Box Testing

It is a test case design method used on the functional requirements of the software. It will help a software engineer to derive sets of input conditions that will exercise all the functional requirements of the program. Black Box testing attempts to find errors in the following categories

1. Incorrect or missing functions.
2. Interface errors.
3. Errors in data structures.
4. Performance errors.

The game has been exhaustively tested by official Nokia QA Team at Nokia Quality Testing Center, Canada. We have tested the game locally on Nokia C7-00 and Nokia N8 devices successfully.

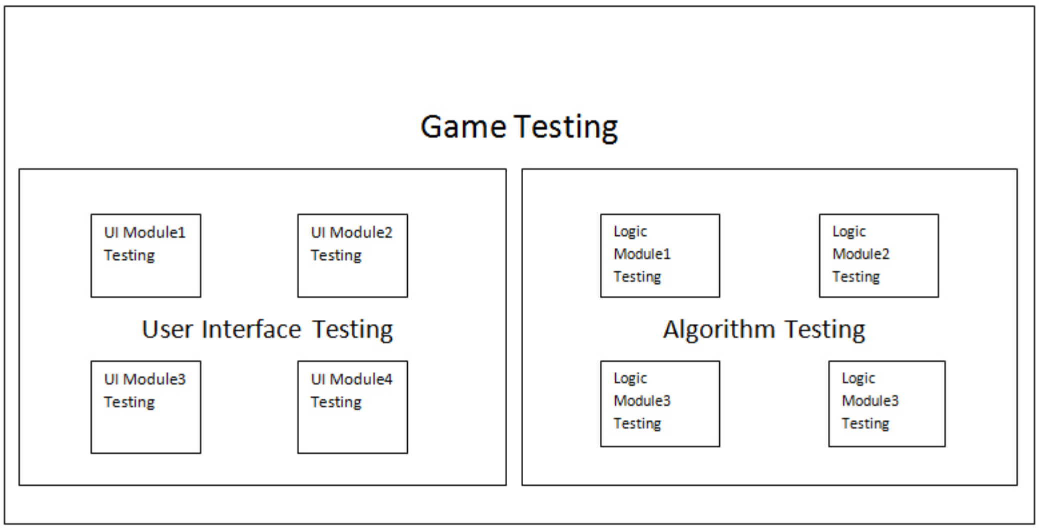
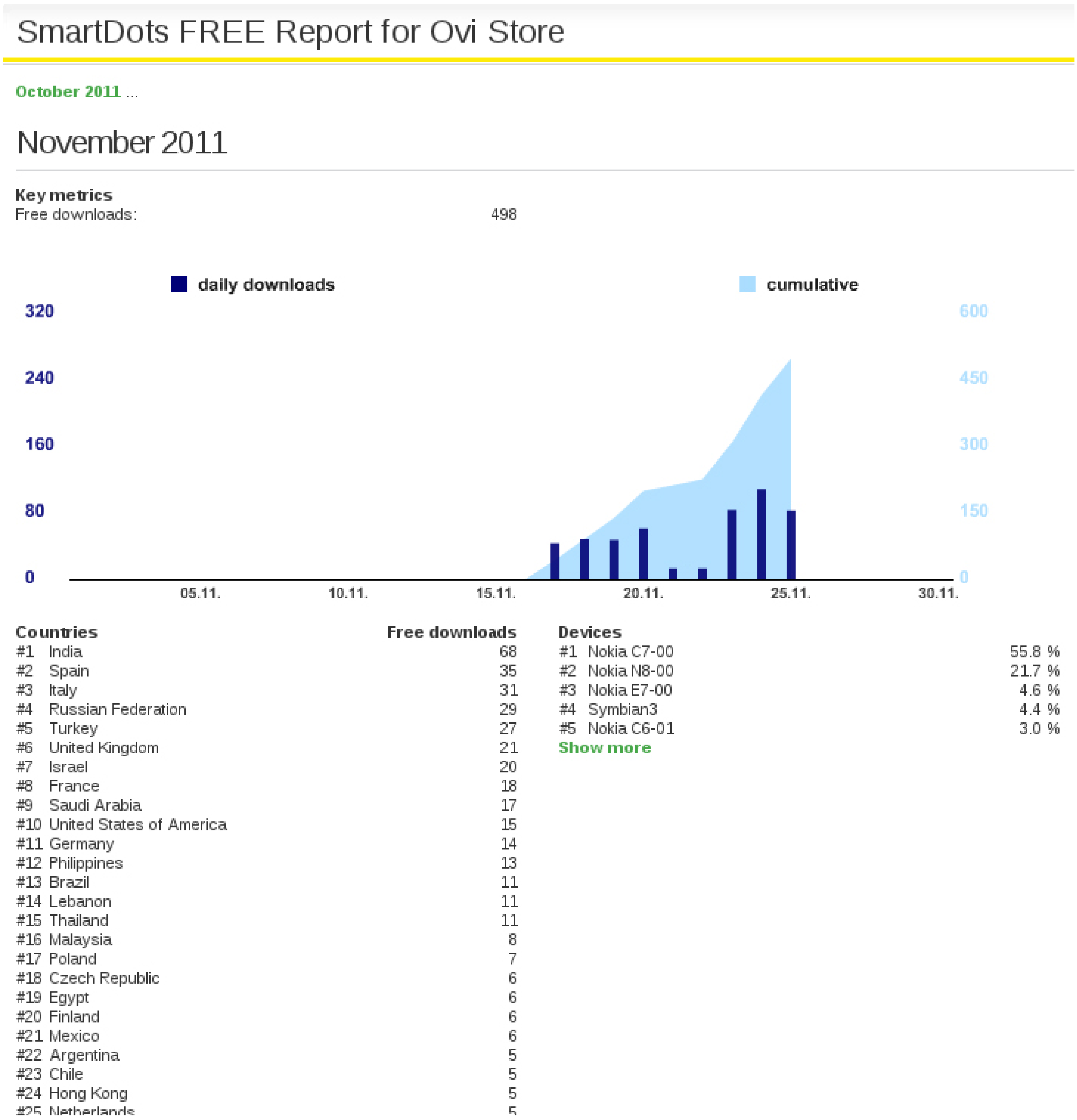


Figure 5: Modularized testing of Algorithm and User Interface

11 Conclusion

The mobile application is thoroughly and satisfactorily complete in all aspects of game development. After rigorous testing at the Nokia Testing Center, Canada and by the OVI QA team, the application was published in the Nokia OVI Store on 18/11/2011. It is enjoying a good number of downloads and is one of the top applications for Nokia C7 and Nokia N8 devices on the OVI Store. The application fetched downloads of more than 100 on the first day of its appearance on the OVI Store. It is being downloaded across 75 countries. The application can be downloaded at http://store.ovi.com/content/220031



SmartDots FREE Report for Ovi Store

October 2011 ...

November 2011

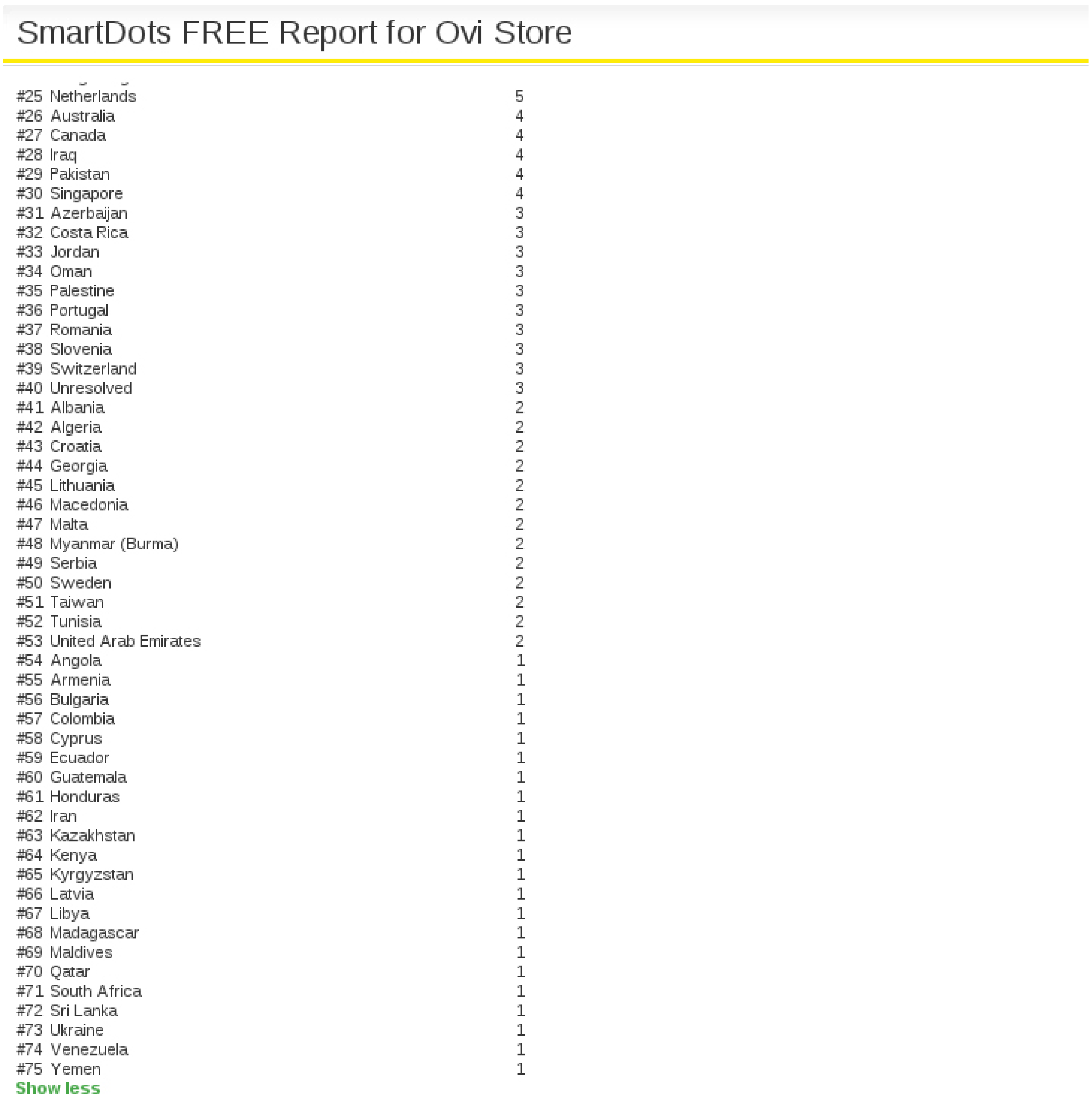
Key metrics
Free downloads: 498

Countries		Free downloads
#1	India	68
#2	Spain	35
#3	Italy	31
#4	Russian Federation	29
#5	Turkey	27
#6	United Kingdom	21
#7	Israel	20
#8	France	18
#9	Saudi Arabia	17
#10	United States of America	15
#11	Germany	14
#12	Philippines	13
#13	Brazil	11
#14	Lebanon	11
#15	Thailand	11
#16	Malaysia	8
#17	Poland	7
#18	Czech Republic	6
#19	Egypt	6
#20	Finland	6
#21	Mexico	6
#22	Argentina	5
#23	Chile	5
#24	Hong Kong	5
#25	Netherlands	5

Devices		
#1	Nokia C7-00	55.8 %
#2	Nokia N8-00	21.7 %
#3	Nokia E7-00	4.6 %
#4	Symbian3	4.4 %
#5	Nokia C6-01	3.0 %

Show more

Figure 6: Screenshot from Nokia OVI store download page



SmartDots FREE Report for Ovi Store

#25	Netherlands	5
#26	Australia	4
#27	Canada	4
#28	Iraq	4
#29	Pakistan	4
#30	Singapore	4
#31	Azerbaijan	3
#32	Costa Rica	3
#33	Jordan	3
#34	Oman	3
#35	Palestine	3
#36	Portugal	3
#37	Romania	3
#38	Slovenia	3
#39	Switzerland	3
#40	Unresolved	3
#41	Albania	2
#42	Algeria	2
#43	Croatia	2
#44	Georgia	2
#45	Lithuania	2
#46	Macedonia	2
#47	Malta	2
#48	Myanmar (Burma)	2
#49	Serbia	2
#50	Sweden	2
#51	Taiwan	2
#52	Tunisia	2
#53	United Arab Emirates	2
#54	Angola	1
#55	Armenia	1
#56	Bulgaria	1
#57	Colombia	1
#58	Cyprus	1
#59	Ecuador	1
#60	Guatemala	1
#61	Honduras	1
#62	Iran	1
#63	Kazakhstan	1
#64	Kenya	1
#65	Kyrgyzstan	1
#66	Latvia	1
#67	Libya	1
#68	Madagascar	1
#69	Maldives	1
#70	Qatar	1
#71	South Africa	1
#72	Sri Lanka	1
#73	Ukraine	1
#74	Venezuela	1
#75	Yemen	1

Show less

Figure 7: Screenshot from Nokia OVI store downloads page contd.

12 Future Enhancement

The future enhancements of the game are as follows

1. As the KDE is entirely built on Qt, this game will be ported to Kubuntu, Fedora and many other Linux distributions as a desktop application.

2. Background music will be added to enhance the player experience of the game.

3. Portability of the game will be increased by modifying the screen size to fit most Nokia mobile phones.

4. As per the reviews from the user who downloaded the game from Nokia Ovi Store, appropriate modifications and enhancements will be done.

13 Screenshots of SmartDots

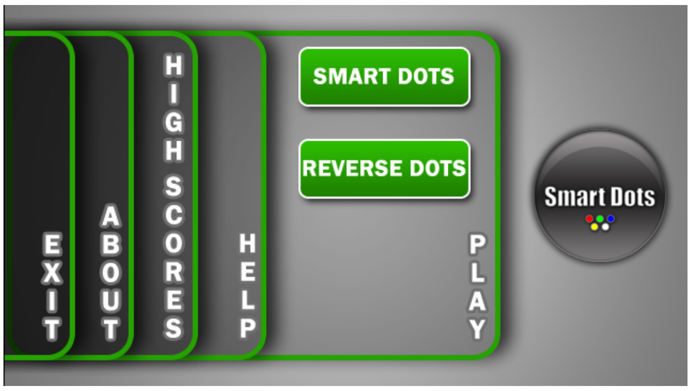


Figure 8: Play menu

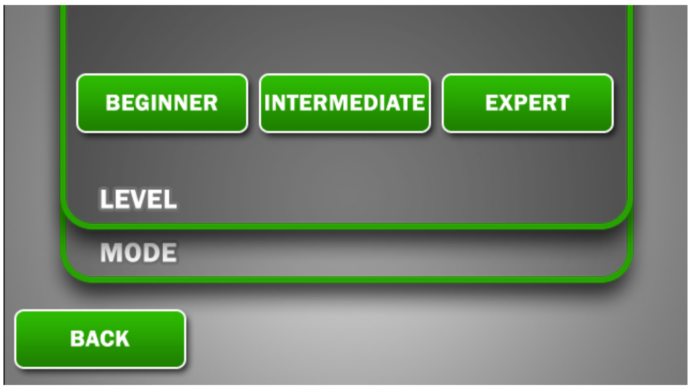


Figure 9: Level select menu

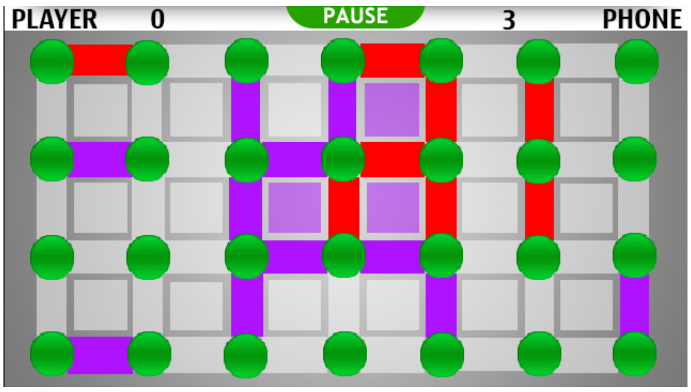


Figure 10: Play area

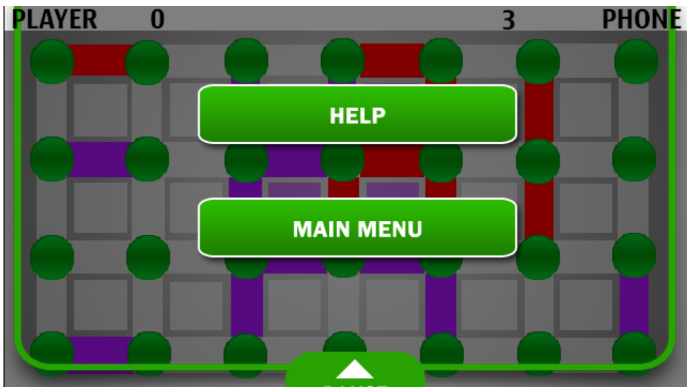
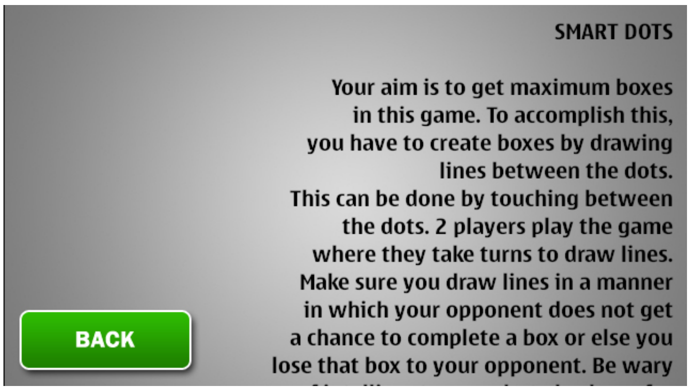


Figure 11: Pause menu



SMART DOTS

Your aim is to get maximum boxes
in this game. To accomplish this,
you have to create boxes by drawing
lines between the dots.
This can be done by touching between
the dots. 2 players play the game
where they take turns to draw lines.
Make sure you draw lines in a manner
in which your opponent does not get
a chance to complete a box or else you
lose that box to your opponent. Be wary

Figure 12: Help view

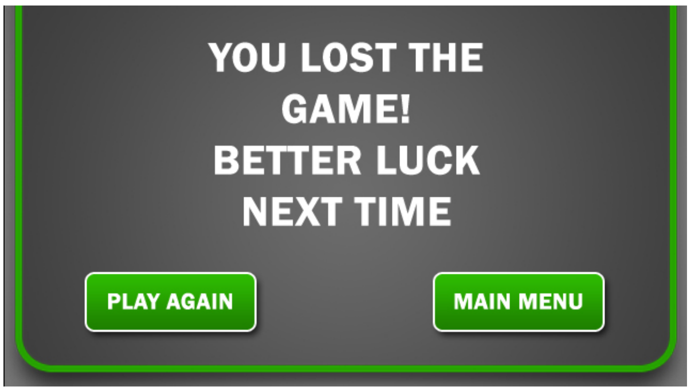

YOU LOST THE
GAME!
BETTER LUCK
NEXT TIME

Figure 13: Result view

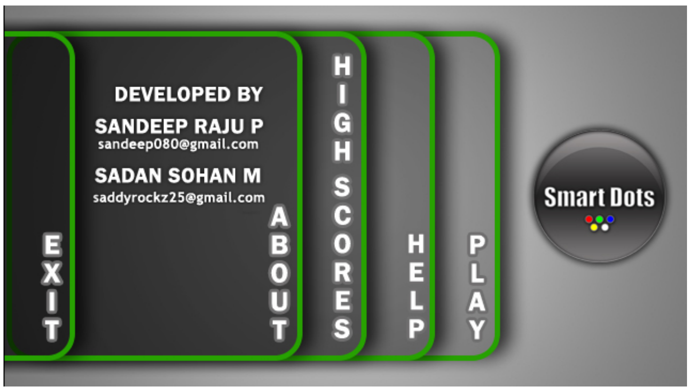

DEVELOPED BY
SANDEEP RAJU P
sandeep080@gmail.com
SADAN SOHAN M
saddyrockz25@gmail.com

Figure 14: About view