

GAME DESIGN IN EDUCATION

Mark Overmars
Institute of Information and Computing Sciences
Utrecht University
3584 CH Utrecht, The Netherlands
E-mail: markov@cs.uu.nl

KEYWORDS

Game design, Education, Game Maker.

ABSTRACT

Computer games play a very important role in the life of most youth. Games offer many possibilities in education. Various people have studied the use of existing games or specially designed educational games. In this paper we consider the use of game design itself in an educational context, with a focus on high schools. Game design can be used in many different subjects, ranging from languages and arts, to mathematics and computer sciences. We also introduce the *Game Maker* software package that can be used by students to easily create their games.

INTRODUCTION

Most kids, students and young adults love playing computer games. It has become an important part of their life. They are willing to spend vast amounts of time on improving their skills for playing these games and the span of attention they have for these game is very long. This is a clear reason for educators to investigate how games can be use in education. This has led to a range of educational games, the formation of a sub discipline of game design, named *serious gaming*, and an increasing number of researchers studying these phenomena and their implications for the education systems, see e.g. (Prensky 2001).



Figure 1. Rome Total War can be used to obtain insight in the way the Roman Empire expanded itself.

One approach is to use standard games in an educational context. Games like *SimCity* can create understanding of economic systems, *Super Monkey Ball* can teach kids about certain physics principles, and *Roman Total War* can give historic insight in the Roman Empire (see Figure 1). Unfortunately such games are not tuned toward the educational

practice and, hence, are not easy to employ in the classroom. Some teachers also use games to reward kids for their work but that seems a poor approach.

The more common approach is to create games that are particularly written for certain educational goals. Unfortunately, most of these games are of poor quality. There are many reasons for this. One is that the budgets for educational games are normally orders of magnitude smaller than those for normal games. As a result, kids will easily be disappointed with the result. But more importantly, game principles and educational goals are often conflicting. Without going in detail, there is on one hand the educational demand that often wants to put the control over what is learned in the hands of the teacher, while a crucial ingredient of interesting game play is that the player should have control over the action. While games can be good in providing insight, letting the player explore some domain, understanding intricate mechanisms, and making motivated choices, most educational games focus on learning specific facts and skills without providing an adequate motivation within the game world. There is still a long way to go before the educational system has changed such that games will play an important role there.

In this paper I want to explore a third way of using games in an educational context. Rather than using games I want to focus on the creation of games. Creating games requires a large number of skills which can easily relate to certain subjects in schools. Also, creating a game about some topic can be a good and motivating way of understanding certain material. Game design can be used in language education, arts education, computer science, physics, geography, and many other areas. I will indicate some of these possibilities and then concentrate on the *Game Maker* program that I wrote for this.

ASPECTS OF GAME DESIGN

Creating a computer games involves many different aspects. The game play must be defined, the story must be written, the characters must be designed, levels must be created, and interaction and behavior of computer controlled entities must be programmed. Usability tests are required to make sure the game satisfies the player's demands, and a marketing and promotion plan is required to actually sell the game. All these aspects can be used in an educational context.

Language classes for example could study the important aspects of stories in games. See for example (Glassner 2004) for an overview of the role of stories in games. Students could write a game story, for example for an adventure game. This would be highly motivating, in particular when their stories would be used in other classes to be turned into an actual game.

In art classes, rather than e.g. drawing portraits, students could design game characters that express a certain archetype, or they can paint game backgrounds that add the correct atmosphere to the game. They can design the 2-dimensional or 3-dimensional levels in

which the game takes place. And, going a step further, they could design animations and even introductory movies for the games.

In music classes students could study game music and the effect it has on the player. They could investigate adaptive music that smoothly adapts itself to the game situation and they could compose their own game music using the various packages available.

MATHEMATICS

The core of a game consists of the rules that define the game play. Rules on one hand describe the inner mechanisms inside the game (for example how much damage a particular weapon does to an enemy) and on the other hand describe the moves the player can make. See (Salen and Zimmerman 2004) for an extensive study of rules in games. This applies equally well to board games, sport games, and computer games. A prime aspect of designing a good game is to come up with a set of rules that is consistent, balanced, and meaningful. This obviously requires creativity but also mathematical skills, in particular in logic and probability calculus.

Such systems of rules are very suitable for an educational context. For example, students could investigate the well-known rock-paper-scissors principle in which paper defeats rock, rock defeats scissors, and scissors defeats paper. Such a simple cyclic relation unfortunately leads to rather boring gameplay (the best strategy is to make random choices) but more complex relations can be studied as well, like the one depicted in Figure 2 that is rather common in real-time strategy games (Rollings and Morris 2003). Here students could deduce that for example infantry is not required to win a game. Schemes can be made more interesting by adding costs to entities (often called shadow costs in game design terminology) and calculations can be made to determine the correct relative costs between units. This leads to various aspects of the mathematical field of game theory. See e.g. (Dutta 2001) for an easy introduction.

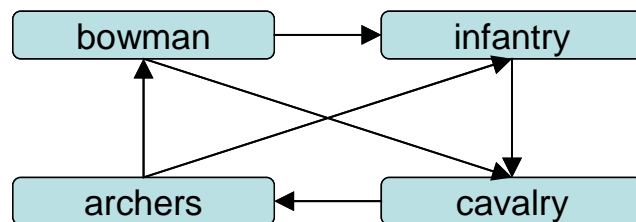


Figure 2. The relation between different units in a strategy game. An arrow indicated the relative strength, for example, the archers are stronger than the (horse) bowman.

Probability also plays an important role in games. Many games contain events that happen with a certain chance, for example the appearance of a special bonus. The game designer must choose such chances correctly to obtain interesting game play. This can be used to introduce probability theory to students. For example, students could be asked to design some sort of casino game in which they have to determine the correct height of the prices in relation to the chance that certain combinations of dice or playing cards appear. Even letting students design a simple game involving a few dice already leads to a fun challenge (see Figure 3).

COMPUTER SCIENCE

When I was taking my first steps in programming I was very excited when I could write a program to compute the first 100 prime numbers. Nowadays, computer applications provide access to music, video, and games, and novice programmers want to create similar programs. Unfortunately, such programs are difficult to create. Teachers have tried to raise student interest by using languages such as

as *Logo* (www.logosurvey.co.uk) that can

create interesting drawings, or robots, such as the *Lego MindStorms* (www.legomindstorms.com). But using Logo to make drawings is no longer flashy enough, and robots are rather expensive and limited in their functionality.



Figure 3. Some high school students, during a workshop, are designing a game with some dice and colored stones.

Creating computer games on the other hand is a challenge that many students want to take on. Developing computer games involves many aspects of computing, including computer graphics, artificial intelligence, human-computer interaction, security, distributed programming, simulation, and software engineering. It can be used as a vehicle to teach students about these topics.

Many teachers indicate that it is difficult for students to understand object-oriented programming. This is somewhat surprising because object-oriented design is very natural. In real-life we think in terms of objects with certain properties and behavior. Still, once people write a program they tend to adopt the traditional view of instructions being executed and control structures.

But when you are creating computer games, the situation changes. In a computer game, everything is an object: the monsters, wall segments, coins, bonuses, power-ups, and the guns and bullets. Thinking about creating games means thinking about objects and how they react to one another and to the player's input. So the game creator naturally thinks in an object-oriented way.

Also inheritance, a powerful but sometime difficult to grasp object-oriented programming concept, becomes much easier to understand in a game design context. Take, for example, the well-known class of games based on *Breakout*, in which the user must destroy stones by hitting them with a bouncing ball. All stones exhibit similar behavior but will appear in a variety of shapes and colors. These characteristics make it logical and efficient to create one stone object and specify its behavior, then create similar objects with different colors that inherit the original stone object's behavior. Making modified

stones that override certain behavior also becomes an easy concept. For a more extended description see (Overmars 2004).

Finally games can be used to let students understand how to plan and execute a larger project. Creating games can involve different people (even from different courses) that must work together as a team. Writing design documents, performing usability studies, and setting up a good testing environment are crucial for the success of games.

GAME PROJECTS

Game design can also play a role in other subjects. For example, in areas like physics and chemistry students could create games to investigate or explain concepts like gravity, electricity, or chemical reactions. In economics they could design their own economic system for a simulated city. And in geography they could make games about the location of certain towns. With a bit of imagination you can come up with many more possibilities. In this way students can create their own educational games.

GAME MAKER

Creating computer games is not an easy process. Commercial games are the combined work of teams of 10 to 50 people and require budgets of millions of dollars. And even creating smaller, simpler games from scratch would be a complicated task requiring advanced programming skills. Fortunately, there are a number of software packages available that make it way easier to create computer games, replacing (part of) the programming by mechanism in which games are constructed from simple building block. Examples are *StageCast* (www.stagecast.com), that is particularly aimed at young kids, and the products by *ClickTeam* (www.clickteam.com). Many similar packages exist, several of which can be found at www.ambrosine.com/resource.html.

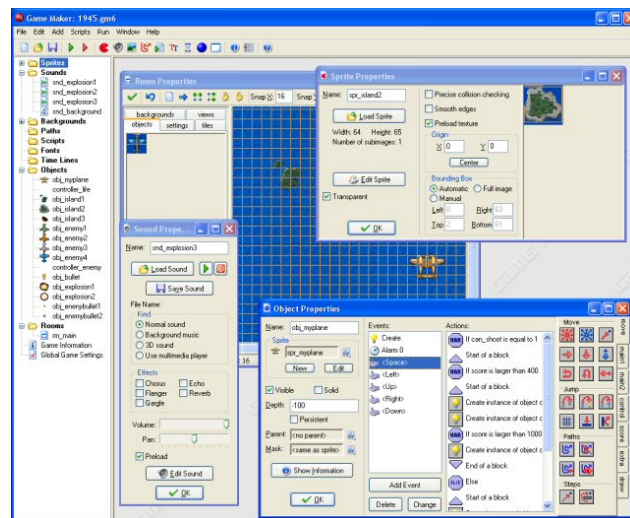


Figure 4. The Game Maker interface. The left side displays resources such as sprites and sounds, while the right side shows the object editor, room editor, and other property forms.

In this paper we will concentrate on *Game Maker* (www.gamemaker.nl), written by the author. Game Maker, is a rapid-application development tool currently used worldwide by young people at home and in schools to create two-dimensional and isometric games.

Figure 4 shows the Game Maker interface, which uses an object-oriented, event-driven approach. With Game Maker's drag-and-drop techniques, users can create games without writing a single line of code, but it also includes an interpreted programming language. The program produces stand-alone games that can be distributed freely; a version of Game Maker itself is available for free as well.

Game Maker has become extremely popular during the past few years. In 2003, users downloaded over a million copies of the program. An active user community exists with many forums (see <http://forums.gamemaker.nl>). The youngest users, 8-year-olds, receive their introduction to computer programming through Game Maker. The oldest users are 80-year-old senior citizens.

Game Maker uses an object-oriented design concept as described above. Creating a game consists of defining objects. Some objects have a visual representation, such as an animated sprite. Others, like those that control game flow or maintain the score, might lack this feature. Multiple instances of the same object can appear in the game at the same moment.

Instances have properties. Some are built-in, like the speed with which the instance moves and the sprite used to represent it. Others can be defined, manipulated, and checked using actions or code. The user must define each object's behavior. While some objects, like wall segments, will have no behavior, others, like the avatar representing the player, will most likely have complicated behavior.

Game Maker defines behavior in event-driven terms. Events occur for objects, and the designer specifies actions that the game must execute when these events occur. Typical events include object creation or destruction, user input, collisions between instances, and alarm clocks. To achieve this, the game designer can simply drag and drop actions into events, as Figure 5 shows. Inheritance is achieved by simply setting the Parent field in an object. This indicates that behavior is inherited from another (parent) object.

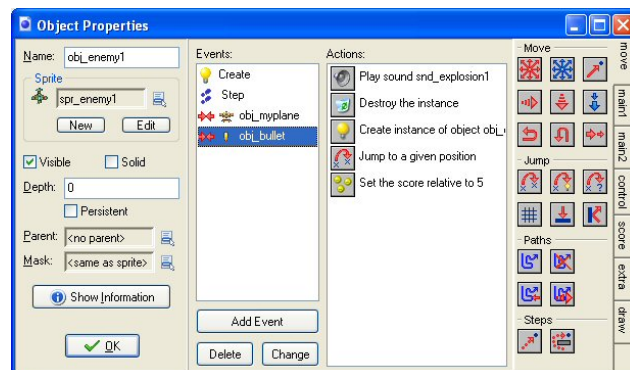


Figure 5. Object property form. The list of defined events for the enemy appears in the left center, while the actions that the game must perform when the enemy collides with a bullet appear to the right.

Game Maker has more than 100 built-in actions, ranging from moving the object in a particular direction to playing a sound or displaying a high-score list. For more advanced tasks, the designer uses a code action to type in pieces of code that are executed when the event occurs. Within this code are close to 1,000 possible functions that can control all aspects of the game, including a particle system, network play functionality, and routines

for 3D graphics. Students in general will start using the drag-and-drop actions but soon realize the use of writing pieces of code. In this way they are naturally introduced into the concept of programming.

Once the objects are defined (and the required sprites and sounds are added to the game) the designer can create rooms (or levels) using the room editor. Instances of objects are placed in the rooms and when the game is executed these instances come to life because of the actions in their creation events and they start reacting to each other and to the user input through the actions in their collision events and keyboard or mouse events.

Creating games with Game Maker is very efficient. After some experience with the program, a typical *Pacman* clone takes less than an hour to create. As a result the students can concentrate more on the design aspects of the games rather than on all the details of getting the game to work. People have created all sorts of games with Game Maker, ranging from simple maze games and scrolling shooters to adventure games and strategy games. See Figure 6 for some examples of their creations.

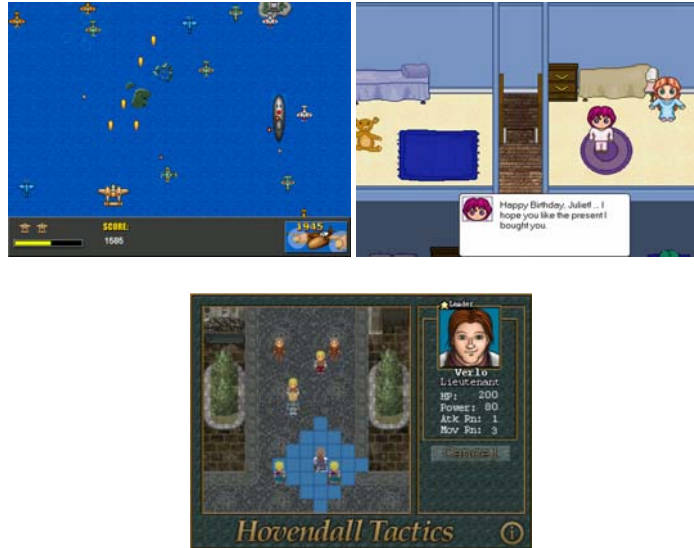


Figure 6. Some games created with Game Maker.

CONCLUSIONS

Creating games appeals to all ages and to both males and females. It involves a lot more than programming, bringing together aspects of liberal arts, mathematics, social sciences and computer science. By using game design in an educational context you create an enthusiastic group of students that are eager to learn and who will find out that creating games can be even more fun than playing them.

REFERENCES

- Dutta, P.K., 2001. *Strategies and Games*, The MIT Press.
- Glassner, A., 2004. *Interactive Storytelling: Techniques for 21st Century Fiction*, A K Peters, Ltd.
- Overmars, M.H., 2004. "Teaching computer science through game design." *IEEE Computer* 37, no.4 (Apr): 81-83.
- Prensky, M., 2001. *Digital Game-Based Learning*, McGraw-Hill .
- Rollings A. and D. Morris, 2003. *Game Architecture and Design*, New Riders Publishing.
- Salen K. and E. Zimmerman, 2004. *Rules of Play, Game Design Fundamentals*, The MIT Press.