Development of Cognitive Video Games for Children with Attention and Memory Impairment

by

David William Bartle
B.Sc., University of Victoria, 2010

A Thesis Submitted in Partial Fulfillment of the
Requirements for the Degree of

MASTER OF SCIENCE

in the Department of Computer Science

Development of Cognitive Video Games for Children with Attention and Memory Impairment

by

David William Bartle
B.Sc., University of Victoria, 2010

Supervisory Committee

Dr. Bruce Gooch, Supervisor
(Department of Computer Science)

Dr. Amy Gooch, Departmental Member
(Department of Computer Science)

Dr. Yvonne Coady, Departmental Member
(Department of Computer Science)

Supervisory Committee

Dr. Bruce Gooch, Supervisor
(Department of Computer Science)

Dr. Amy Gooch, Departmental Member
(Department of Computer Science)

Dr. Yvonne Coady, Departmental Member
(Department of Computer Science)

Abstract

Children with Fetal Alcohol Spectrum Disorder (FASD) may suffer numerous cognitive impairments, including significant problems with executive functioning, language, attention, and memory [40]. It is estimated that two to five percent of children born in the U.S. are affected by FASD [34]. It has been shown that training improvements can be made in working memory and attention in children with ADHD [25]. Computerized training with game elements enhances not only motivation but training efficacy of these interventions [38]. This thesis examines the creation of two suites of serious games, *Cognitive Carnival* and *Caribbean Quest*, intended to improve working memory aspects of attention with the assistance of a trained psychology interventionist in a therapeutic setting. A game-based approach is chosen to provide motivation to children for sustained cognitive challenges presented by cognitive exercises built into the gameplay. *Cognitive Carnival* was shown by interventionists to have positive effects in neuropsychological studies of populations of children with epilepsy and FASD [33, 30].

Contents

Supervisory Committee **ii**

Abstract **iii**

Table of Contents **iv**

List of Tables **vii**

List of Figures **viii**

Acknowledgements **ix**

Dedication **x**

1 Overview **1**

1.1 Introduction 1

1.2 Background 1

1.2.1 Serious Games 1

1.2.2 Cognitive Remediation 2

1.2.3 Definitions 3

1.2.4 Medical Conditions 4

1.3 Objectives of the Thesis 5

1.3.1 Part I: Objectives for *Cognitive Carnival* 5

1.3.2 Part II: Objectives for *Caribbean Quest* 6

2 *Cognitive Carnival*: A Game-based Cognitive Rehabilitation Tool **8**

2.1 Introduction 8

2.2 Game Design Principles 9

2.2.1 Negative and positive feedback 9

2.2.2 Motivation, encouragement, and engagement 9

2.2.3 Accommodating physical limitations . . . 9
2.3 Game Design Challenges . . . 10
2.3.1 Duration of Materials . . . 10
2.3.2 Difficulty . . . 10
2.4 Implementation of *Cognitive Carnival* . . . 11
2.4.1 Software platform . . . 11
2.4.2 Working memory . . . 12
2.4.3 Sustained attention . . . 15

3 Evaluation of *Cognitive Carnival* 18
3.1 Overview . . . 18
3.2 Pilot Study . . . 19
3.3 FASD Studies . . . 20
3.3.1 First study . . . 20
3.3.2 Second study . . . 20
3.4 Lessons Learned from *Cognitive Carnival* . . . 21
3.4.1 Reward systems . . . 21
3.4.2 Duration of materials . . . 22
3.4.3 Theme . . . 22
3.4.4 Progress . . . 22
3.4.5 Enjoyment . . . 22
3.4.6 User experience . . . 23
3.4.7 Working memory string generation . . . 23
3.4.8 Miscellaneous . . . 23
3.5 Summary . . . 24

4 *Caribbean Quest*: A Web-Delivered Intervention 25
4.1 Introduction . . . 25
4.1.1 Challenges in delivery of interventions . . . 25
4.2 Design of Caribbean Quest . . . 26
4.2.1 Distinction of cognitive and game parameters . . . 26
4.2.2 Describing arguments with XML . . . 27
4.3 Implementation of *Caribbean Quest* . . . 28
4.3.1 Software platform . . . 28
4.3.2 *NewCola*: a Silverlight game engine . . . 29

4.4 Working Memory Games 30
4.4.1 Working memory parameters 30
4.4.2 Generating working memory spans 32
4.5 Dynamic Difficulty 34
4.6 Games of Caribbean Quest 35
4.6.1 Scuba 35
4.6.2 Pirate Delicatessen 36
4.7 Sustained Attention Games 37
4.7.1 Submarine 38
4.7.2 Wave 39
4.8 Selective Attention Games 39
4.8.1 Squidditch 40
4.9 Motivational Structure 41
4.9.1 Bonus games and the Sand Dollar currency 41
4.9.2 The Overworld 45
4.9.3 Store 46
4.10 Summary 48

5 Conclusions and Future Work **49**
5.1 Conclusions 49
5.1.1 *Cognitive Carnival* 49
5.1.2 Caribbean Quest 50
5.1.3 Final Remarks 51
5.2 Future Work 51
5.2.1 Parameters 51
5.2.2 Progress 52
5.2.3 Measuring and controlling difficulty 52

Bibliography **54**

A Parameters and Source Code Listings **59**
A.1 Parameters from *Cognitive Carnival* 59
A.2 Parameters from *Caribbean Quest* 62
A.2.1 Game parameters 62
A.2.2 Cognitive parameters 64
A.2.3 Backtracking algorithm for working memory sequence generation 66

List of Tables

Table 4.1 Working memory attributes. 33

Table A.1 *Wheel* parameters. 59
Table A.2 *Platform* parameters. 60
Table A.3 *Liftoff* parameters. 61
Table A.4 *Scuba* Parameters. 62
Table A.5 *Pirate Delicatessen* parameters. 62
Table A.6 *Submarine* parameters. 63
Table A.7 *Wave* parameters. 63
Table A.8 Working memory parameters. 64
Table A.9 Continuous performance parameters. 65
Table A.10 Visual scanning parameters 65

List of Figures

Figure 2.1 The *Liftoff* game from *Cognitive Carnival*. 13
Figure 2.2 The *Platform* game from *Cognitive Carnival* 15
Figure 2.3 The *Wheel* game from *Cognitive Carnival*. 17

Figure 4.1 Class hierarchy of cognitive parameters in *Caribbean Quest*. . 27
Figure 4.2 Class hierarchy of game parameters in *Caribbean Quest*. . . . 27
Figure 4.3 A typical XML argument file 28
Figure 4.4 Class view of working memory attributes. 34
Figure 4.5 The *Scuba* game from *Caribbean Quest*. 36
Figure 4.6 The *Pirate Delicatessen* game from *Caribbean Quest*. 37
Figure 4.7 The *Submarine* game from *Caribbean Quest*. 38
Figure 4.8 The *Wave* game from *Caribbean Quest*. 39
Figure 4.9 The *Squidditch* game from *Caribbean Quest*. 40
Figure 4.10 The *Scuba* bonus game from *Caribbean Quest*. 42
Figure 4.11 The *Submarine* bonus game from *Caribbean Quest*. 43
Figure 4.12 The *Wave* bonus game from *Caribbean Quest*. 44
Figure 4.13 The *Cannon* bonus game from *Caribbean Quest*. 45
Figure 4.14 Part of the Overworld from *Caribbean Quest* 46
Figure 4.15 The same part of the Overworld from Figure 4.14. 47
Figure 4.16 The Store from *Caribbean Quest* 48

Acknowledgements

I thank, in no particular order:

NSERC USRA, GRAND, and NeuroDevNet, for funding portions of the development of *Cognitive Carnival* and *Caribbean Quest*,

Google, Inc. and Lime Connect for the 2011 Google Lime scholarship,

Jennifer MacSween, for design, sound, and QA for *Cognitive Carnival* and *Caribbean Quest*,

Jeremy Long and Sam Rossoff, for assistance in implementing *Cognitive Carnival*,

Jason Cummer, Angela Jeske, Tim Jordison, and Rob Kelly, for software development work on *Caribbean Quest*,

David Baumgart, Karen Campbell, Weston Roda, and Matthew Steele, for art, sound, and music contributions for *Caribbean Quest*,

Greg King and Lindsey McDowell, for QA on *Caribbean Quest* and editing contributions to this thesis,

Kennedy Denys, Jacqueline Pei, and Roxanne Vernsecu for intervention design and feedback for the games,

Yvonne Coady, for making me choose computer science,

Amy Gooch, Bruce Gooch, and Kim Kerns, for providing this opportunity,

and my parents, for providing many others.

They'll fix you. They fix everything.
Robocop (1987)

Dedication

To Bruce Creswick (with apologies to T.S. Eliot):
you were the music while the music lasted.

Chapter 1

Overview

1.1 Introduction

Children with Fetal Alcohol Spectrum Disorder (FASD) may suffer numerous cognitive impairments, including significant problems with executive functioning, language, attention, and memory [40]. It is estimated that two to five percent of children born in the US are affected by FASD [34]. Training has shown to improve working memory and attention in children with ADHD [25] and furthermore, computerized working memory training with game elements has shown to enhance motivation *and* efficacy [38]. This thesis examines the creation of two suites of serious games (*Cognitive Carnival*, and *Caribbean Quest*), intended to improve working memory and sustained attention abilities with the assistance of a trained psychology interventionist in a therapeutic setting.

1.2 Background

1.2.1 Serious Games

Serious games are a type of interactive entertainment where beneficial qualities such as education, training, or rehabilitation are the primary goal. Serious games can provide structure and motivation to users to aid in accomplishing objectives and to make tasks more engaging. Serious games predate computers and video games; one of the earliest known examples of what is now referred to as a serious game is *Kriegsspiel* [31], which was used to train Prussian army officers in war strategy as early as 1812. The tradition has continued to present times: the United States Army has demonstrated a strong

interest in serious games, including *America's Army* [35], a series of online multiplayer first-person shooter games that serve as recruitment tools by emphasizing positive aspects of the U.S. Army, especially teamwork. Researchers have explored educational games ("edutainment") and their effectiveness as learning tools in educational settings [24, 17] as well as specifically using games for teaching [22]. The serious game *FoldIt* [14], a game based on a realistic simulation of protein folding, reverses the typical role of serious games by having users perform tasks that ultimately help its creators, by finding optimal solutions using game motivation techniques. *FoldIt* awards points based on the plausibility that a fold is likely to be correct. The candidate folding solutions can then be examined by lab testing.

Serious games have also been used for health and rehabilitation. Commercial video games like Nintendo's *Super Mario Bros.* [36] have been used in psychology case studies on children to examine their approach to problem-solving and ability to cope with success and failure [23]. Twillert et al. [50] demonstrated that immersive virtual reality environments can help mitigate pain and anxiety for burn victims when changing dressings. Rizzo et al. [41] modified the commercial title *Full Spectrum Warrior* [45] to include a set of five customizable virtual reality scenarios for assisting veterans from Operation Iraqi Freedom with Post Traumatic Stress Disorder (PTSD). Most recently, *Robomemo* (part of CogMed™) has been used to demonstrate improvements in working memory [15].

1.2.2 Cognitive Remediation

Training has shown to improve working memory and attention in children with ADHD [25] and other disorders including brain injury [46]. Specifically, cognitive remediation approaches aimed at strengthening underlying attention and working memory abilities by performing repetitive tasks that uniquely exercise components of those abilities, and gradually increasing the attention demand, processing speed and cognitive load.

The theoretical basis for such an approach, from the work of Luria [32], suggests that the brain displays a high degree of plasticity and that functional reorganization of neural pathways can facilitate improvements in cognitive function. Indeed, evidence from neuroscience demonstrates that direct, process-specific training can enhance underlying cortical activity.

1.2.3 Definitions

As this thesis is in the area of computer science, but contains discussion of neuropsychological interventions, it is important to define some essential terms from outside the domain of computer science that will be used frequently for the remainder of the text. The games of this thesis focus primarily on three cognitive functions: working memory, sustained attention, and selective attention.

Working Memory

Definition 1.1. ***Working memory*** *is the brain system which actively holds information in the mind to do verbal and nonverbal tasks such as reasoning and comprehension, and to make it available for further information processing. Working memory tasks are those that require goal-oriented active monitoring or manipulation of information or behaviours in the face of interfering processes and distractions.*

The working memory tasks in the games of this thesis typically present the user with a sequence of 3–7 distinguishable objects. These sequences of objects are referred to interchangeably as *working memory spans*, *working memory strings*, and *working memory sequences*.

Sustained Attention and Selective Attention

In 1890, the American psychologist William James described attention thus:

> *"Everyone knows what attention is. It is taking possession of the mind in clear and vivid form of one out of what seem several simultaneous objects or trains of thought."* [28]

Since James, there have been many definitions of attention put forward, all defining attention as a multidimensional construct including components such as sustained attention, selective attention, vigilance, inhibitory control and visual search.

Definition 1.2. *Within the model used to conceptualize attention training materials, two specific aspects of attention are targeted. I refer to these as* ***selective attention*** *and* ***sustained attention****.*

1.2.4 Medical Conditions

The software reported in this thesis was designed to be utilized for children with either epilepsy or FASD. An overview is included to establish the context of the software design.

Fetal Alcohol Spectrum Disorder (FASD)

Fetal Alcohol Spectrum Disorder (FASD) is an umbrella term for a group of disorders present in individuals that is caused by their mother's ingestion of alcohol during pregnancy. Alcohol is a teratogen–a substance that is toxic to the developing brain–and can result in abnormal brain development and a broad spectrum of difficulties with several aspects of cognition and behaviour. Children with FASD may suffer numerous cognitive impairments, including significant problems with executive functioning, language, attention, and memory [40].

There is a high degree of variability in the severity of cognitive disability seen in children with FASD, from individuals suffering from general intellectual deficiency to those with deficits in more specific skills, such as language [13]. It is estimated that two to five percent of children born in the US are affected by FASD [34]. This software was created to study the utility of therapies that help restore abilities in individuals with brain damage, and measure its efficacy for these individuals. As such, the research methods focus on rehabilitation and particular exercises for the brain.

Epilepsy

According to the American Academy of Neurology, epilepsy is a "brain disorder in which clusters of nerve cells, or neurons, in the brain sometimes signal abnormally" [**?**]. Many studies have found that cognitive impairment is a common finding in patients with epilepsy. It is unclear why these impairments occur, whether it is due to the seizure activity itself, the side effects of medications used to control the seizures, or the as-yet-unknown cause of epilepsy. All have been theorized to be the cause of the impaired cognition associated with the disorder.

> Children with epilepsy are at significant risk for problems with inattention, hyperactivity and impulsivity. 20% of children with epilepsy meet full diagnostic criteria for attention deficit hyperactivity disorder (ADHD) [47].

> Epilepsy is associated with decreased short-term and working memory (WM) skills [9]. Multiple studies have documented increased incidence of learning difficulties and underachievement in the areas of reading, writing and mathematics[19]. [33]

1.3 Objectives of the Thesis

This thesis concerns the design, implementation, and evaluation of two suites of serious games targeting working memory, sustained attention, and selective attention. The objectives of the thesis are thus split into two parts: those determined before the initial design of *Cognitive Carnival*, and those discovered after its completion and field use, incorporated into the sequel *Caribbean Quest*.

1.3.1 Part I: Objectives for *Cognitive Carnival*

The development of *Cognitive Carnival* was an iterative interdisciplinary collaborative effort. Psychologists with expertise of working memory, sustained attention, and selective attention provided training exercises that formed the basis for the specification and design of the software. My primary objective was to translate as much of this expertise as possible into the software:

Objective 1.1. *Integrate working memory, sustained attention, and selective attention training material into video games.*

To achieve this, the training materials must be expressed in terms of "parameters" (collections of data types that control the aspects of the cognitive difficulty).

Objective 1.2. *Compile the working memory, sustained attention, and selected attention training options that control the cognitive difficulty into collections of concrete data types, or "parameters".*

By compiling training exercises into concrete data types, we can express the training in software, and therefore in a video game. The larger goal of *evaluating* the therapeutic qualities of the games required the software to be used by interventionists in a study, outside the scope of the thesis but summarized in Chapter 3.

1.3.2 Part II: Objectives for *Caribbean Quest*

Upon the conclusion of the *Cognitive Carnival* interventions, I determined new objectives for the cognitive games. These are incorporated into the follow-up software, *Caribbean Quest* (Chapter 4).

Objective 1.3. *Design a software component to dynamically adjust the difficulty of the working memory games to accommodate each participant's current performance.*

The working memory games in *Cognitive Carnival* proved to be too easy for some participants and too difficult for others. This compromised the effectiveness of the study to some degree. To accommodate differing abilities, a software component is necessary to adjust the working memory difficulty on a per-user basis.

Objective 1.4. *Determine the software relationship between the games and the cognitive components of difficulty.*

The games of *Cognitive Carnival* did not share any code, but after iterating there were (in hindsight) clearly intersecting requirements. This indicated that the cognitive (working memory, sustained attention, etc.) and game (e.g. enemies, level geometry) elements of difficulty should be separated into distinct modular components, for better code maintenance and the ability to directly contrast cognitive difficulty across different games.

Objective 1.5. *Find a way to make the amount of materials adjustable and keep motivation high, while still having distinct goals and rewards.*

The traditional game structure of a set of levels leading to a final objective does not apply well to an intervention setting where the amount of time is fixed, as opposed to playing the game for however much time it takes to reach the end, then stopping. Having distinct goals, however, is important to maintain engagement and motivation. Therefore the games need a design consideration and/or system for allowing both lower- and higher- achieving players to accomplish goals and obtain rewards.

Objective 1.6. *Allow a flexible and expressive way to describe the game parameters, independently of the game binary.*

Objective 1.2 establishes cognitive parameters represented in data types. In *Cognitive Carnival*, the contents of the levels were integrated directly into the game

binary and fixed for the duration of the study. The ability to modify the levels during the study is important as it is difficult to estimate the difficulty within levels and from one level to the next, without empirical results. To allow interventionists direct control of the therapy, they need the ability to modify the arguments that compose the different levels of the games.

Objective 1.7. *Create a robust algorithm for generating working memory strings, and a uniform interface to access it.*

Objective 1.4 attempts to establish distinct game and cognitive difficulty components. Provided this separation, each working memory game can share the working memory parameters. To reduce redundant code, a general working memory generator system is a logical conclusion. With a uniform interface, multiple games could access it. Furthermore, a more robust algorithm that could account for any number of attributes, and allowed to easily express concepts like whether an object is "nameable" or is distinct from others could not only reduce the likelihood of errors, but allow the therapy to be expanded with images and concepts not yet explored in its initial configuration.

Chapter 2

Cognitive Carnival: A Game-based Cognitive Rehabilitation Tool

2.1 Introduction

Chapter 1 established that serious games have been successfully applied in rehabilitative settings; that cognitive remediation has been shown to be effective at improving memory and attention in individuals with brain damage, as well as children with various developmental disorders (e.g. ADHD); and that game elements can improve motivation and efficacy in such training. This suggests that a serious game incorporating memory and attention training could provide a compelling and efficacious environment to children in an intervention.

Working with a team of psychologists, I developed *Cognitive Carnival*, a suite of serious games aimed at promoting the exercise of specific cognitive abilities ("cognitive games") [7]. The primary objective of *Cognitive Carnival* was to provide children an engaging environment in which to work with an interventionist, including hierarchical structure, goals, and rewards. *Cognitive Carnival* was used in three neuropsychological studies detailed in Chapter 3.

2.2 Game Design Principles

The games in *Cognitive Carnival* were nonviolent and appropriate for children as young as five years. The primary study was a population of children with FASD (See section 3.3.1), ergo the games were tailored to incorporate design considerations to overcome limitations outside of memory and attention, in order to make the software as accessible as possible. These principles also underly the sequel *Caribbean Quest* (Chapter 4).

2.2.1 Negative and positive feedback

I wished to minimize potential sources of discouragement within the games. Therefore, negative feedback was avoided in the games, beyond the minimum requirement of indicating whether a task had succeeded or failed. Furthermore, common game elements such as lives and health were not included in the design of the games.

Contrariwise, positive feedback was added to the game wherever possible to provide encouragement and motivation to the participants. Players were rewarded with points for every completed task, and trophies for completing major goals. Positive sounds were played when a participant completed a level to reinforce success.

2.2.2 Motivation, encouragement, and engagement

Interventions are hard work. The participants must complete difficult objectives to succeed. For children–especially children with behavioural issues–motivation can be fleeting. To increase engagement in an intervention, typically the participants will be rewarded with extrinsic real-world items such as toys or vouchers. I wished for the games to have as much *intrinsic* motivation as possible. To this aim, I added "trophies" to the game, which were awarded when certain objectives were completed. The participant can view all of the trophies he or she has earned in a dedicated "Hall of Fame" screen in the game.

2.2.3 Accommodating physical limitations

Children with FASD may have impaired motor skills [29]. This can severely inhibit their ability to play video games that require dextrous movements and coordination. I wanted the games to provide an environment for motivation and enjoyment, not

to hinder a participant's ability to perform the cognitive tasks, so I endeavoured to minimize the amount of fine control required to succeed..

2.3 Game Design Challenges

The games of *Cognitive Carnival* presented two interrelated fundamental design challenges: the duration of materials and the difficulty of the tasks.

2.3.1 Duration of Materials

Given the literature on this type of intervention, the objective was to have an appropriate amount of material so that the participants could complete the training in about 12 hours of play time, presented in 30-minute sessions over five weeks; this stated, given the wide variation in abilities between children in their cognitive abilities, it was clear that it would not be possible to accurately estimate the time for all participants to complete the tasks [30]. Moreover, although the variation in materials and content was well established, the interaction of the parameters–especially in working memory tasks–does not have a well-described model of difficulty (discussed further in Section 2.3.2).

2.3.2 Difficulty

Although it is intuitive that longer sustained attention tasks are more difficult than shorter ones, and likewise with longer vs. shorter working memory spans, there are also more subtle issues of difficulty, especially for working memory spans. The working memory spans are made of a discrete number of objects. With the meaningful range of spans being about three to seven objects, there are only a handful of options available. Each increase in span length represents a large increase in difficulty, from very easy to supremely difficult, which prevents the possibility of a gradual increase in difficulty as the levels progress.

To increase difficulty beyond the span length, there is a wide array of options that can be varied to produce different tasks. For example, participants may be asked to recall a span in reverse order, requiring them to remember the individual items in order to manipulate them. Other variations on working memory tasks include the presence or absence of visual or auditory distractions (e.g., stimuli which

are unrelated to the task at hand), whether there are items within a presented list to be ignored (e.g., a list of numbers in which you must repeat back only the odd numbers), whether the list of items is presented visually or verbally, and whether you are responding by recognizing the list from an array of presented and non-presented elements. The configuration of these options is referred to as *working memory parameters* throughout this thesis. In *Cognitive Carnival*, these parameters are part of the *Liftoff* and *Platform* parameters (Tables A.3 and A.2).

Examples of working memory parameters include whether the sequence has distracting elements, or requires sorting. There is a combinatorial explosion of possibilities. While it is clear in the literature that tasks requiring manipulation of information in mind are more difficult than those that just require maintenance of information, I was unable to find any comprehensive models for studies examining precisely what increases the difficulty of a given working memory task. Given that for cognitive remediation tasks, the principle of presentation of training materials is done in a hierarchical order for difficulty, this lack of a model made it impossible to determine which combinations of parameters are more difficult than others (and this is assuming that there isn't a particularly large variance between parameter combinations in participants).

Cognitive Carnival did not address this challenge in the design of the software, relying on a static difficulty curve instead, which was designed by my collaborators. The importance of a better understanding of difficulty emerged after the software was used in studies; the increased priority of this problem introduced Objective 1.3, which was included in the design of *Caribbean Quest*.

2.4 Implementation of *Cognitive Carnival*

2.4.1 Software platform

I implemented *Cognitive Carnival* using the Microsoft® XNA™ framework [4] and Microsoft .NET framework [2] in the C# programming language [5]. The games were built for Microsoft Windows®. XNA provides the programmer with an API and libraries for creating games, and also has integration with the Xbox 360™ [6] controller; the controller helps provide children with a more game-like interface, while avoiding the fine control required with a keyboard.

Cognitive Carnival is composed of three mini-games: *Liftoff*, *Platform*, and *Wheel*.

While the games are primarily focused on either working memory or sustained attention–and are categorized as such–all of the games contain aspects of both.

2.4.2 Working memory

Liftoff

The *Liftoff* game, at its simplest, is a *Simon*-type [10] game, where the goal is to store, manipulate, and repeat a sequence of objects by selecting them with an on-screen cursor. The sequence is either shown visually or presented orally to the user. The objects of the sequence have multiple properties (e.g. colour, shape, number), which I will refer to as "tokens". Since the tokens in *Liftoff* have multiple properties, many possibilities of rules are available, such as "ignore shapes that are purple". These options provide more subtle ways of adjusting the task difficulty than changing the span length alone.

How the sequence is presented, how it is recalled, and a myriad of distractions, along with the basic length of the sequence, create the different levels and distinguish it from the popular handheld electronic toy from the 1980's. The participant must recall sequences which are either shown visually or presented orally. Some levels of the game require the user to ignore extra distracting elements, or to sort the elements using a particular strategy (e.g. alphabetizing) instead of simply maintaining the list and repeating it. The full *Liftoff* parameters are listed in Table A.3.

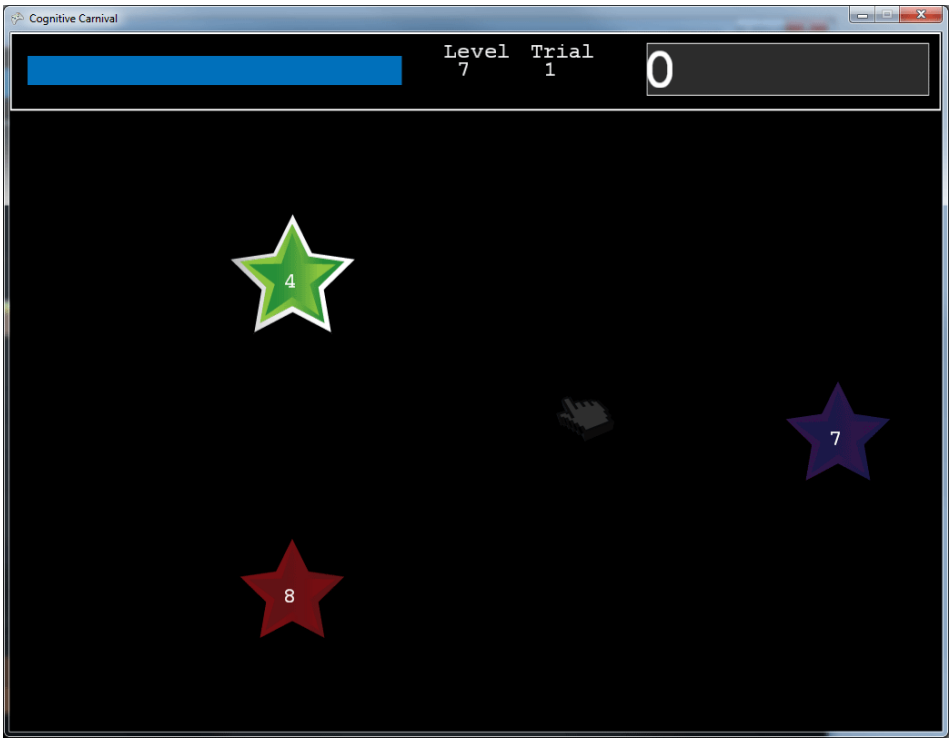

Figure 2.1: The *Liftoff* game from *Cognitive Carnival*. In this level, the tokens have differing colours and numbers, but all are the same shape (stars). The participant must repeat the sequence in reverse order. The game is in the process of presenting the sequence visually; currently the green star is being highlighted. When the presentation is complete, the cursor at the centre of the screen will be enabled.

Platform

The *Platform* game, as its name suggests, is a 2D platforming game where the player navigates an avatar up and down ladders, avoids enemies, and jumps over hazards, similar to *Lode Runner* (1983) [16]. These components make *Platform* the most dexterous game of the suite. The participant is required to remember a random sequence of objects of various types, collect them in the specified order, then navigate to an exit door. The participant must retain the sequence while being distracted by the other elements of the game, such as enemy characters. The *Platform* game tends to offer simpler working memory tasks than *Liftoff*, the other working memory game, to account for the increased amount of game difficulty. For example, the objects are always collected in forward or reverse order, with no sorting.

The *Platform* game employs both visual and auditory instructions to present the

working memory span to the participant. There are several categories of objects for the player to collect, including fruit, sports balls, and flags of the world. The types of objects can greatly affect the difficulty of remembering the items: for example, the fruit category is composed of objects that children encounter in everyday life, like apples and oranges. The sports category includes similarly familiar objects like baseballs and soccer balls, but has a few obscure entries like the whiffle ball, with which several participants in the studies (Chapter 3) were not familiar. This caused an observed drop in performance when the item would appear in a task. As a last example, the flags of the world were mostly incomprehensible to the children, as they were not familiar with them and were hard to distinguish from one another. I expressed this difficulty characteristic with the "nameable" parameter. The full *Platform* parameters are listed in Table A.2.

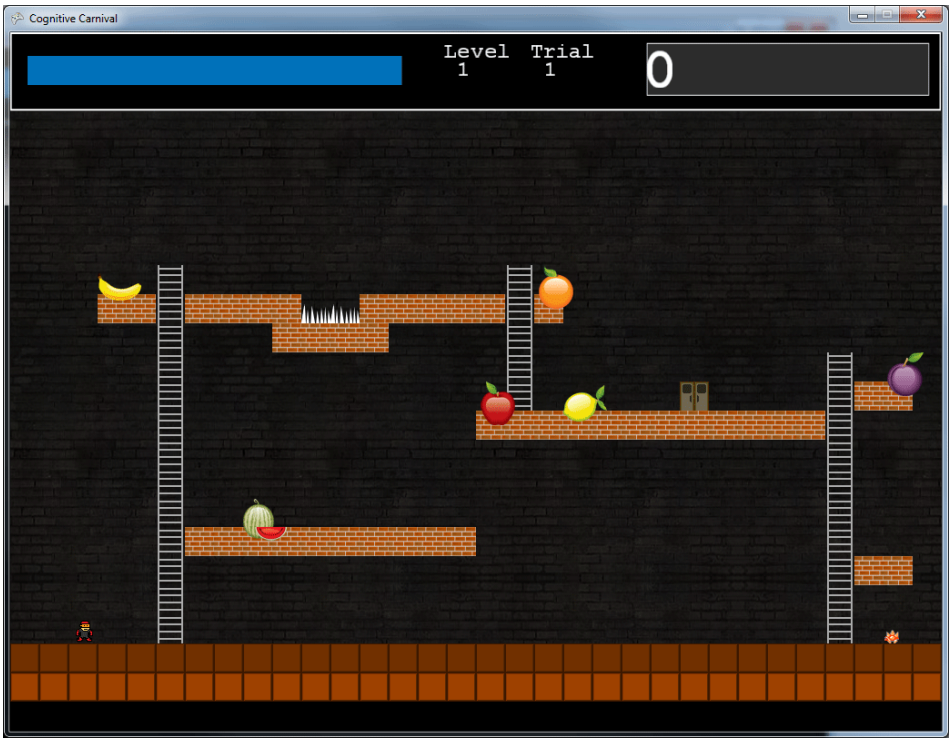

Figure 2.2: The *Platform* game from *Cognitive Carnival*. The player avatar is in the bottom-left corner. The category of objects is fruit, and the player must remember to collect a banana, orange, and apple in that order. The plum, lemon, and watermelon are distracting elements. In the bottom-right corner there is an enemy which will end the round if touched. Finally, on the platform in the middle-right portion of the screen is the exit door. The door will open when the sequence has been repeated correctly.

2.4.3 Sustained attention

Wheel

In the *Wheel* game, a large wheel with symbols on it rotates, causing the symbols to pass through a selection box. If the user presses a button, the symbol currently in the selection box is activated. The participant is given rules at the start of the level indicating which symbols need to be selected. Symbols include various coloured shapes and animals. Sustained attention is the key ability that is necessary to complete this task. Since the wheel itself is entirely fixed and only requires player input to select an item, *Wheel* requires minimal dexterity. If the user selects a target that they were

not supposed to, or fails to select a target that was part of the rule, that symbol will glow red, indicating an error. Each level has a set duration; when the time expires, the participant's correctness percentage is calculated. The participant must exceed a certain accuracy to proceed to the next level, otherwise the level must be repeated.

The early *Wheel* levels begin with a simple task, such as remembering to select a specified stimulus (e.g. a red star). The participant must remember to select that object each time, ignoring other symbols. The task requires the participants to stay vigilant for a short duration of time (e.g. two minutes), and increases steadily with each level to a maximum of six minutes. By gradually increasing the length of the continuous performance task, the participant gradually increases their sustained attention span. On later levels, the task rules increase in complexity; for example, one possible rule could be "select stars, but only if you see a red square right before it."

The *Wheel* game optionally has distractors in the form of images that periodically appear and disappear on the edges of the screen. Their frequency of appearance and proximity from the edges of the screen can be adjusted to provide various levels of distraction, requiring higher levels of selective attention, as the participant is required to ignore all task-irrelevant information. The full set of *Wheel* parameters are listed in Table A.1.

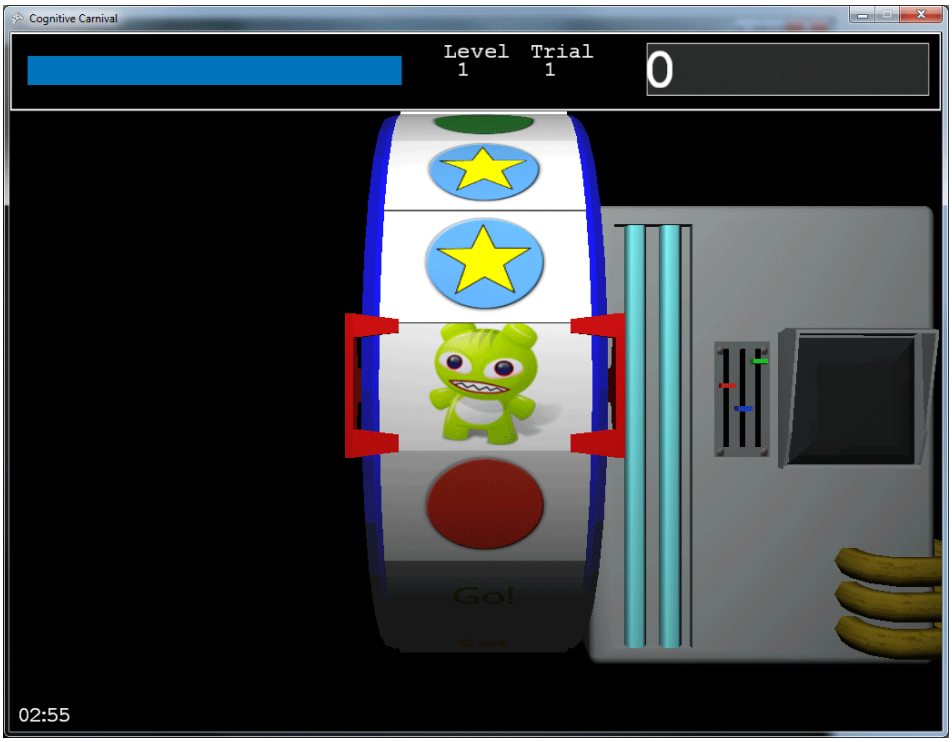

Figure 2.3: The *Wheel* game from *Cognitive Carnival*. In this task, the user must select any yellow stars as they pass through the red selection box *if* they appear immediately after a green bear. This level lasts three minutes and requires an accuracy of 90% to advance.

Chapter 3

Evaluation of *Cognitive Carnival*

Cognitive Carnival was used in three studies. The deeper neuropsychological aspects of the studies are beyond the scope of this thesis, and moreover are outside of the area of expertise of the author. The implications of the results, however, are important to establish the success or failure of the software as a rehabilitative tool, and are therefore summarized in this chapter.

3.1 Overview

The first was a pilot study which took place in the Educational Psychology Department at the University of Victoria, Victoria, British Columbia, as part of a program in which children with epilepsy were seen once a week after school to participate in tutoring or other activities to improve their school performance. The participants were all children between the ages of six and thirteen years who were diagnosed with epilepsy and having difficulty in school. The second and third studies investigating the impact of this cognitive intervention on children with FASD was conducted through the University of Alberta in Edmonton, Alberta.

The participants for these studies were all children diagnosed with FASD between the ages of six and sixteen. These children were all provided with the intervention at school during their regular school hours.. There were a total of 39 participants between the two studies.

3.2 Pilot Study

Given the difficulty of identifying children with FASD in the school districts in Victoria, prior to developing *Cognitive Carnival* the investigators were aware that the FASD studies would need to occur off-campus. As such, a pilot study with children with epilepsy offered a chance to get the software into the hands of children with difficulty in this area, and allowed the investigative team to observe early results and issues in game performance. As noted, a group of eight children between the ages 6–13, all diagnosed with epilepsy, were included in this pilot study. The participants played 30 minutes of the *Cognitive Carnival* intervention once a week over the course of 12 weeks, for a total of six hours of intervention time. Children played the game working with a trained "interventionist" who worked with the children providing support and motivation, metacognitive strategies, and scaffolding their performance on the game.

Prior to the intervention, children completed a number of measures of aspects of attention from the Test of Attentional Performance in Children (KITAP, which includes tasks measuring distractibility, divided attention, sustained attention) [53]. As well, children completed standardized measures of reading and math fluency, and tests of attention and working memory (spatial and digit span tasks from an intelligence test). The same tasks were administered post-intervention and statistical comparisons were conducted to investigate the efficacy of *Cognitive Carnival*.

On the KITAP attention tasks, following the intervention participants made significantly fewer errors on both the distractibility task, as well as a sustained attention task. In addition, they showed longer spatial span backwards, and significant improvements in reading and math fluency.

Following the pilot study, I modified *Cognitive Carnival* to allow for more flexibility in setting the difficulty level of the working memory tasks on a per-user basis, since some users found the initial tasks too difficult to complete, and others found them too simple. This early pilot work with *Cognitive Carnival* set the stage for the need for full dynamic difficulty support (Objective 1.3), which would eventually be developed for the *Caribbean Quest* sequel (Section 4.5).

3.3 FASD Studies

3.3.1 First study

Participants for the first study were recruited through the Edmonton Public School Board from March to June, 2010. Eighteen children diagnosed with FASD between the ages of six and twelve participated in this randomized control trial. Each child received 24 half-hour sessions over a 12-week span, participating in approximately two sessions per week. The children were split into two groups, one receiving the intervention condition (*Cognitive Carnival*, abbreviated "CC" for brevity throughout the rest of this chapter) and the other receiving a control condition (playing computer games focusing on educational materials but not attention or working memory training).

During the intervention, the students played the computer games with the aid of an interventionist who helped the students apply metacognitive and self-regulation skills, while the interventionists themselves executed the scaffolding techniques, tailoring their supports to each individual students needs and abilities, then gradually reducing them as the students competence increased (fading).

As with the epilepsy study, standardized measures of attention, working memory and academic skills were given before and after the *Cognitive Carnival* intervention. Both the children who received CC and the control group were compared on these measures following the intervention. Results of the study revealed that both groups made some significant improvements on several of the standardized measures. In further analysis of the data, it was noted that while participants were randomly assigned to the two groups, the control group started out with better skills in general than did the children completing the CC in intervention.

3.3.2 Second study

The final study was also completed with children recruited through the Edmonton Public School Board. Twenty-one children diagnosed with FASD between the ages of six and sixteen participated in this study. The design of the study differed from the earlier pilot, and utilized a 'waitlist' control approach where children were randomly assigned to either the active intervention group (who started the intervention immediately following post-testing), or a wait list control group (who started the intervention 13 weeks after their initial enrolment and pre-intervention assessment. Children in

both groups received an initial assessment of several measures of attention, working memory, and academic abilities.

All children were tested again at 12 weeks. This served as a post-test assessment for the children in the immediate intervention group. This assessment in the waitlist control group allowed the psychologists to determine how much change in performance these children would exhibit after 12 weeks without any active intervention. While data for this study are still being analyzed, preliminary results revealed significant improvements on a standardized measure of attention and working memory, with significant improvements on percentage of total correct responses.

3.4 Lessons Learned from *Cognitive Carnival*

Cognitive Carnival successfully demonstrated that cognitive games can offer enhanced motivation and–provided professional support–can, over the course of weeks, lead to small but significant improvements in some psychological measures in children with epilepsy or FASD. Feedback from participants and interventionists indicated potential areas of enhancement in the games. I attempted to address each of these concerns in the design of the eventual sequel, *Caribbean Quest* (Chapter 4).

3.4.1 Reward systems

Research interventionists reported that participants were excited to be rewarded with trophies, and that these features provided significant motivation to continue playing. As these are fairly trivial to add to games, they are an efficient way of increasing motivation and positive feedback.

The participants were confused by not knowing when the trophies would be awarded. Within *Cognitive Carnival*, there are two reasons for this: first, there was no explicit breakdown or visual indication of how much work was required to earn a reward; second, I only had a vague idea of how much time was required to achieve each goal. As such, it was not clear how best to implement the rewards. But, given the experience gained from the studies, I focused on improving the reward system with a few new design principles; by giving participants a way of measuring their progress towards achievements, we can provide them with medium-term goals. Since the games are played in 30-minute sessions, it is logical to design the games to require about this much time to obtain an in-game reward. This establishes Objective 1.5.

3.4.2 Duration of materials

The research interventionists noted significant decreases in motivation when participants completed the games at a certain cognitive demand (working memory span) and then had to start the same series again at a higher level of difficulty. Since the theoretical basis of the intervention is repetition of the cognitively-demanding tasks within the game, with a gradual increase in cognitive load, a certain number of hours of intervention training not an arbitrary end of the material to gain enough intervention hours participants started the games over at a higher difficulty level. The games in *Cognitive Carnival* were not designed to be completed more than once; they did not provide any additional goals for a subsequent playthrough. All possible rewards were already exhausted. Therefore, it is not surprising that the participants were discouraged after reaching the end. This establishes Objective 1.5.

3.4.3 Theme

The games were developed iteratively, adding difficulty parameters for experimentation. Unfortunately, one of the side-effects of this development process was difficulty staying on theme. The original "carnival games" theme was largely watered down by the time the games were finished. Furthermore, the games did not share design characteristics beyond their Heads Up Display (HUD), and there was nothing past the main menu screen of the game to connect them.

3.4.4 Progress

Within a game level, *Cognitive Carnival* provided the participants information within a session with a progress bar. This allowed participants to know when they were approaching their short-term goal of reaching the end of a game session. On a larger scale, however, each game simply indicated the participant's current level number; the software did not provide indications for how many levels each game had, or when rewards could be expected. This prevented the participant from measuring their progress towards medium- and long-term goals.

3.4.5 Enjoyment

The primary objective of *Cognitive Carnival* was to ensure that the games provided a large battery of cognitive challenges, sufficient to serve effectively as a rehabilitative

tool in an intervention setting. Although the games provided some motivation and rewards, there were no purely fun elements to *Cognitive Carnival* that are commonly found in typical edutainment titles, such as bonus games. These types of activities can provide an instant reward and greatly increase motivation as participants work towards getting to play the bonus game.

3.4.6 User experience

Cognitive Carnival required menus, dialogues, and other standard user interface elements. Unfortunately, XNA does not offer these controls, and a large amount of development time was spent recreating them. Since the games and parameters were iteratively developed, the user interface also went through many iterations, causing a multiplying effect in the amount of work required.

3.4.7 Working memory string generation

The algorithms required to generate some of the more complex working memory spans and continuous attention tasks are surprisingly nuanced and structured in a specific hierarchical manner. Clearly, their correctness is vital to the successful function of the games, as any failure by the algorithm precludes success for the participant. Field use revealed subtle bugs that can occur with the random and complicated nature of some of the sequence generation code. Furthermore, in hindsight, the working memory games had some intersecting requirements. I concluded that the working memory generation should be more carefully considered, to allow for not only robustness, but predictability of failure and success, as well as a uniform interface to share the functionality among all working memory games. This establishes Objective 1.7 (*Create a robust algorithm for generating working memory strings, and a uniform interface to access it*). The design and implementation of this system are explored in Section 4.4.2.

3.4.8 Miscellaneous

An unexpected side effect of having a 3D perspective game for *Wheel* was that prolonged attention on the spinning wheel actually caused some participants to report mild dizziness.

3.5 Summary

Cognitive Carnival successfully incorporated aspects of working memory, selective attention, and sustained attention into a video game environment. The games were used in three studies, and the interventionists' findings indicate small but significant improvements from comparisons of pre- and post-test measures. Feedback from participants and interventionists indicated that the suite could benefit from additional features, such as the ability to measure one's progress visually.

Chapter 4

Caribbean Quest: A Web-Delivered Intervention

4.1 Introduction

Cognitive Carnival achieved the goal of producing a set of working memory and attention games that were effective in an intervention setting. Based upon feedback from the interventions and researchers, and again in collaboration with the psychology team, *Caribbean Quest* was created to expand the functionality and address some of the shortcomings of the first game suite. The primary goals of the sequel were easier deployment and dynamic difficulty for working memory tasks, though many other problems were addressed.

4.1.1 Challenges in delivery of interventions

Organizing an intervention for children which is to be played over the course of several weeks is a highly nontrivial undertaking. Children need to be taken out of class to perform the training each week. Since removing them from class further impedes their ability to succeed at school, they are typically taken out of non-academic classes (e.g. art class or gym) to help to mitigate the lost classtime. But for many of these students, these missed classes are their favourite subjects, especially since they are areas where they can achieve without necessarily requiring the same abilities to succeed at, say, mathematics or English.

One of the main requirements of *Caribbean Quest* was to allow delivery to homes. Achieving home delivery via traditional desktop applications, as with *Cognitive Car-*

nival, is simply infeasible given the budget realities of a study. Technical issues, hardware compatibility, and instructions for nontechnical end users were deemed to be too expensive and/or difficult to overcome. Therefore, to reach home users, the games were re-imagined as a web-delivered product.

I designed the program to work with a typical computer found in the home. The modest graphics requirements of the games (2D sprites) meant there was no need for a traditional application stack with full access to 3D hardware. Furthermore, the application only requires the user to visit a web link instead of installing a program, greatly simplifying deployment and the cross-disciplinary development process.

4.2 Design of Caribbean Quest

Caribbean Quest was composed of five cognitive games: *Scuba*, *Submarine*, *Wave*, *Pirate Delicatessen*, and *Squidditch*. *Scuba* and *Pirate Delicatessen* are working memory games; *Submarine* and *Wave* are sustained attention games; and *Squidditch* is a selective attention game.

4.2.1 Distinction of cognitive and game parameters

As with *Cognitive Carnival*, *Caribbean Quest* provided a parameter class for each game (*Liftoff*, *Platform*, and *Wheel*). However, I separated the cognitive parameters and the game parameters into distinct classes to enable common interfaces for describing arguments (Section 4.2.2) and to amalgamate intersecting requirements (e.g. the *Scuba* and *Pirate Delicatessen* games both require sequence lengths). This separation resulted in distinct working memory, sustained attention, and visual scanning parameters. The *Scuba*, *Submarine*, *Wave*, *Deli*, and *Squidditch* games each had game-specific parameters (e.g. *Scuba* has an option for enemies, and *Submarine* can have different combinations of portholes open or closed). This allows for a common interface for generating sequences for the games and to easily use the same task on different games.

The parameter classes all inherit from an abstract parent type to allow for common interfaces within the game for loading and saving data. It may seem counterintuitive that the game parameters in *Caribbean Quest* don't inherit directly from the cognitive parameters, since all of the games appear to have an *is-a* relationship with the cognitive parameter class, and are indeed categorized as such in this thesis. However,

though the games are *primarily* focused on one cognitive ability, there are elements that are shared (e.g. *Submarine*, a sustained attention game, contains some working memory load); therefore, the games should not be constrained to a single cognitive parameter classification.

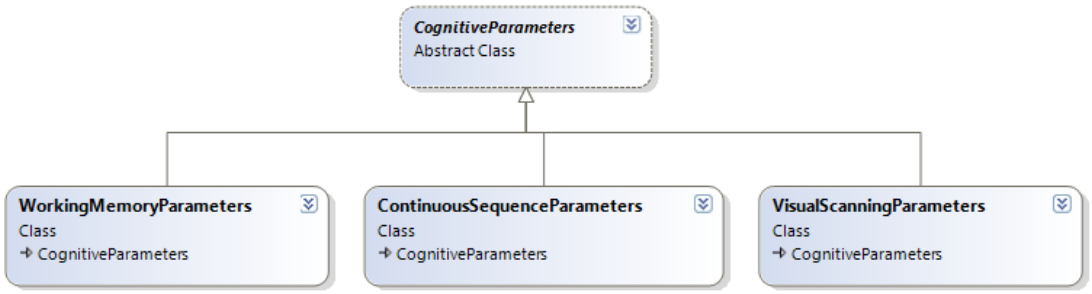

Figure 4.1: Class hierarchy of cognitive parameters in *Caribbean Quest*.

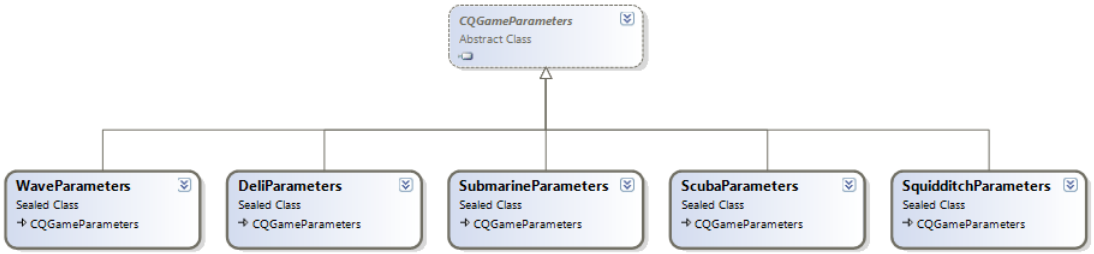

Figure 4.2: Class hierarchy of game parameters in *Caribbean Quest*.

4.2.2 Describing arguments with XML

In addition to separating the parameters into distinct game difficulty and cognitive difficulty types, I used eXtensible Markup Language (XML) [11] to specify the arguments for each game level. I also declared XML schemata [49] for each parameter class. Provided a sensible naming convention, these files should be approachable to a nontechnical user as XML is intended to be "human-legible and reasonably clear" [11]. Furthermore, by using XML, one can leverage existing APIs to parse the data, avoiding the often error-prone and time-consuming task of developing proprietary tools. The separation allows independence between the task content and the game binary: research assistants can modify the tasks on a case-by-case basis–after the software has been deployed–by overriding the default content. In Chapter 5, I propose additions to this part of the software.

```
<?xml version=``1.0'' encoding=``utf-8'' ?>
<WorkingMemoryParameters>
 <length>3</length>
 <auditory>false</auditory>
 <redundantAuditory>false</redundantAuditory>
 <fixedShape>false</fixedShape>
 <fixedColour>true</fixedColour>
 <fixedCharacter>false</fixedCharacter>
 <distinctShapes>true</distinctShapes>
 <distinctColours>false</distinctColours>
 <distinctCharacters>false</distinctCharacters>
 <ignorableColours>0</ignorableColours>
 <ignorableShapes>0</ignorableShapes>
 <ignorableCharacters>0</ignorableCharacters>
 <type>garbage</type>
 <text>none</text>
 <sort>false</sort>
 <reverse>false</reverse>
 <extras>false</extras>
</WorkingMemoryParameters>
```

Figure 4.3: A typical XML argument file from *Caribbean Quest*, describing a working memory task.

4.3 Implementation of *Caribbean Quest*

4.3.1 Software platform

As of this writing there are three popular ways to create games for the web: Adobe®Flash [1], HTML5, and Microsoft Silverlight [3]. Since HTML5 compatibility between browsers–especially for required features such as offline storage–was not finalized at the time of development, I decided to dismiss it as a candidate. I estimated that the scope of the program was large enough that the libraries that accompany the (compact) .NET Framework would benefit the software. After an initial investment of effort into a game engine, the features would show a net gain in development time especially for features such as XML (Section 4.2.2).

4.3.2 *NewCola*: a Silverlight game engine

Game engines are systems that assist in developing video games. Some companies, such as Valve Software, develop their own game engine in-house (the Source engine [43] for *Half-Life 2* (2004) [42], for example); many others license engines for developing games, like Bioware's *Mass Effect* (2007) [8], which uses the Unreal Engine [21].

Clearly, the requirements for *Caribbean Quest* were modest in comparison to multi-million-dollar endeavours. There are, however, several games that have a large potential for overlapping code: the working memory games in *Cognitive Carnival* both shared essentially redundant code for sequence generation, for example, and both of these games have similar analogues in *Caribbean Quest*. I was not aware of any Silverlight game engines that covered all of the technical requirements for the games, so I created *NewCola*, a game engine for Silverlight, which provided a graphics class hierarchy (extended from a top-level abstract *Renderable* class), state management, and several other features. I also designed a *Game* class hierarchy specifically for the *Caribbean Quest* games to share overlapping elements.

Collision handling and physics

Collision handling in video games can range from naive bounding box approaches to realistic physics simulations. Physics engines found in games usually provide a 2D or 3D approximate simulation of rigid bodies and dynamics (the study of forces and motion). Physics objects are updated by the game's update loop and contain physical attributes like velocity, momentum, acceleration, torque, and so on. The game programmer can then render graphics based on the coordinates of the physics objects.

Physics engines are used in many games such as *Crush the Castle* [20] (2009), *Crayon Physics Deluxe* (2009) [39], and *Max and the Magic Marker* (2010) [37]. I decided that the games of *Caribbean Quest* had enough complicated collision requirements (e.g. non-rectangular level geometry, scenes with large numbers of bouncing objects) to justify integrating a physics engine into the games.

I decided to use the Farseer Physics Engine for Silverlight [51] (a .NET port of the widely-used C++ Box2D physics engine [18]) to provide robust collision and dynamics support for the games in 2D. Although most of the games did not rely heavily on physics, it proved to be essential for *Scuba* (in which the coral reef is implemented as a collection of convex polygons), and the *Cannon* bonus game (Figure 4.13), which

leverages the physics simulation to allow obstacles to react realistically[1] to being hit with a cannon ball.

Although using a physics engine provides numerous benefits, it does have overhead both in terms of codebase and computing resources (CPU cycles and memory). I decided that the computing overhead was negligible, but that the codebase overhead could be a problem. To alleviate this, I integrated the physics features directly into the game engine's top-level *Renderable* class.

4.4 Working Memory Games

As in *Cognitive Carnival*, the working memory games in *Caribbean Quest* involve holding a span of information and repeating it according to the given rules. The tasks range in complexity from rudimentary *Simon*-type tasks (repeat the information) to scenarios involving alphabetical and/or reverse sorting, ignoring objects that have certain characteristics, and having additional distracting items on the screen.

Like *Cognitive Carnival*, the working memory games require the user to successfully complete some amount of brief trials to finish a level. Unlike *Cognitive Carnival*, the working memory games of *Caribbean Quest* adjusted the span lengths according to the user's performance with a dynamic difficulty system (Section 4.5), instead of an entirely fixed set of tasks and span lengths.

To provide the user with a way to monitor their progress within a task, I included series of dots corresponding to each task. After each success, a checkmark is placed on the current dot. When all of the dots have checkmarks, the level is complete and the user is taken to a bonus game.

4.4.1 Working memory parameters

The working memory games were the most complex and involved in terms of parameters. Parameters included auditory, distractors, sorting, and several others. The full working memory parameters are listed in Table A.8.

[1]Realism here is used to mean that dynamics are calculated according to Newton's laws, not to imply that shooting a cannonball towards giant gold coins floating in the sky is realistic.

Auditory tasks

All of the objects in the game and their components (e.g. colour) have an associated audio file. These are combined to form auditory instructions. By hearing the sequence instead of seeing it, a much different experience is provided, testing more of the participant's cognitive abilities. Furthermore, redundant auditory information can serve as a distracting element. For example, if a sequence consisted entirely of crates of different colours, telling the user that the object is a crate is unnecessary. If so desired though, this unnecessary information can be included, requiring the participant to filter it and pay attention to the important information. In this way, redundant auditory can serve as a distractor.

Using auditory instructions as the means of communicating the task also allowed for several more parameters to be introduced. For example, the "Blackout" and "Flash" parameters hide the entire sequence, and hide the sequence but display the current elements, while the instructions are playing, respectively. These can be used to increase the difficulty by requiring the participant to store the sequence without being able to immediately correlate the auditory item with the visual on the screen.

Sorting

Simply repeating a sequence back to the computer is fairly straightforward. To increase the working memory load, the sorting parameter requires the user to sort the items and repeat them back in sorted order (e.g. alphabetical), while performing the same gameplay and ignoring distracting elements. Sequences must have numbers or letters for sorting to be applicable.

Distractors

By adding distracting elements ("distractors") to the working memory sequence, we can add to the cognitive load. In the working memory parameters, these distractor elements have two distinct types: "ignorables" and "extras". Ignorables are tokens that are presented to the user as part of a sequence, but the user is instructed to ignore them (e.g. "collect these items except those that are purple in colour"); the user must ignore them entirely when performing the working memory task and remember not to collect these items. Extras are elements that are present on the screen at some point of the presentation, but are not part of the visual or auditory instructions. Precisely

how much these distracting elements interfere with the task is unclear, and beyond the scope of this thesis, but is a question of interest to researchers in this area.

4.4.2 Generating working memory spans

Working memory games require a robust sequence generator for the multitude of different parameters available. To allow both randomness and guaranteed success or failure, I used a backtracking approach. Backtracking algorithms find solutions to problems by iterating through available choices and branching recursively for each choice, forming a tree structure. If a branch causes some specified precondition to become false, the program undoes the choice and navigates back up the tree and continues. Otherwise the algorithm continues branching until it reaches a target depth. This allows for us to exhaust the solution space deterministically and abandon invalid solutions as early as possible, culling large parts of the recursion that would be necessary using a naive method.

Each parameter was represented in the backtracking tree by an "attribute" node. The purpose of attributes is to account for specific properties like colour or shape and record how many elements a parameter supported (for example, the "Digit" attribute supported elements elements, the digits 0–9), and which of these had already been used (if uniqueness was required). The attribute class also provided interfaces for retrieving the resources (e.g. images, sound files) associated with a property. If randomly selecting an index collides with an existing attribute, the generator needs to keep trying to find an available one. To avoid indeterminate running times, instead of generating another random index, the initial random index is incremented, modulus the number of total elements, guaranteeing failure or success after that many iterations.

Each level of the backtracking tree corresponds with an attribute. For example, if we wanted to generate coloured geometric shapes, we could employ a ShapeAttribute and ColourAttribute in succession. By randomly selecting a child node, each element of the working memory span can be constructed. Depending on the requirements of the parameters, the backtracking algorithm accounts for uniqueness and other traits. For example, if a task is auditory, then each element must have at least one property that makes it distinct from the others; otherwise, the task is ambiguous. If the task is visual, this is not strictly necessary.

This approach proves useful when impossible sequences are requested. For exam-

ple, if your attribute was "Fruit Images" and you only had five unique images, and you requested a sequence that is longer than five elements, naive approaches would continue searching for a sixth element indefinitely, since the end is indeterminate. Although one could add conditions to account for such obvious cases, consider a further example: suppose that you had an attribute with a very large size S, and you wanted to pick close to S elements (a valid request). Initially it is likely that you will successfully land on an available index. But as you approach S, the probability of success lowers and the program must search longer to find the last handful of elements. This can cause unpredictable performance in the program, increase the difficulty in finding bugs, and increase the number of CPU cycles. The backtracking approach I designed guarantees success or failure after S operations on a node, and only visits each node once at maximum. Therefore, the maximum time to run the algorithm is predictable and less resources are wasted. The algorithm is listed in Appendix A.2.3.

Attribute	**Description**.
Colour	Primary colour of the object.
Number	Digits 0–9.
Letter	Letters A–Z.
Geometric	Simple shapes (e.g. Square, Circle).
Graphic	Bitmap or vector image.

Table 4.1: Working memory attributes.

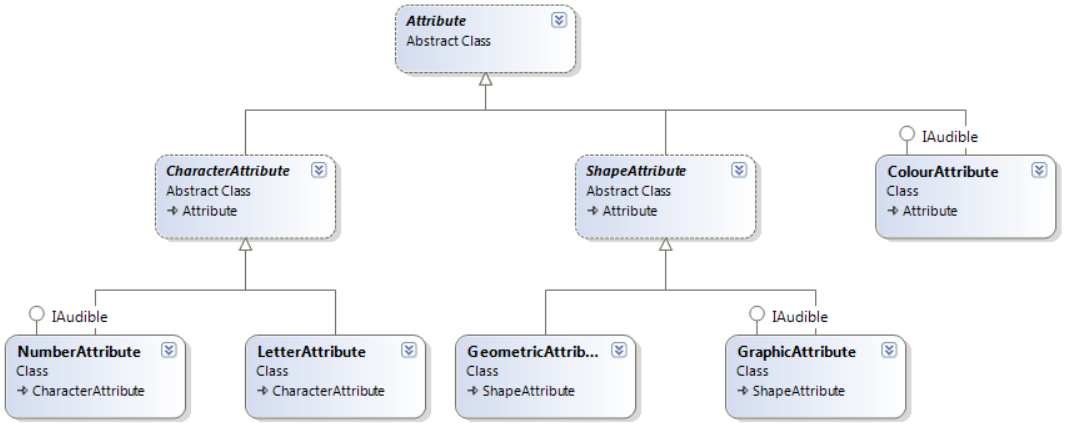

Figure 4.4: Class view of working memory attributes.

4.5 Dynamic Difficulty

Literature on dynamic difficulty in video games tends to revolve around the objective of maximizing enjoyment for players [52, 27, 26]. By giving inexperienced players scaffolding to learn the game, and highly-skilled players additional challenge, the consensus is that dynamic difficulty can accommodate diverse populations of users. Dynamic difficulty can also be used to match differing skill levels between users; Stach et al. [44] adjusted game difficulty to accommodate physical differences in competitive multiplayer exercise games (or "exergames"), to allow for a level playing field. In developing *Caribbean Quest*, the primary motivation was not to increase enjoyment, *per se*. My goal was to dynamically adjust the therapeutic utility of the games, ergo to provide the most appropriate difficulty level for each participant; I wished to avoid frustration for the participant and obtain the greatest possible opportunity for improvement for a fixed amount of intervention time.

Many of the sources of difficulty in video games can be simplified to a single dimension (health bars, number of enemies, etc.) or uniformly across all dimensions (increase all enemies, decrease player health bar), providing unambiguously greater or lesser challenges. But without a well-described model of the relationships between parameters in working memory tasks, there is ambiguity in the difficulty space that we wish to navigate.

As a practical initial effort to accommodate differing user abilities, I implemented

a dynamic difficulty system for the working memory games of *Caribbean Quest* that adjusted span length (an unambiguous source of greater or lesser difficulty). The system adjusts the lengths of the working memory spans according to the user's recent performance, on a per-level basis. The system operates on a per-level basis because each set of parameters has its own *characteristic* difficulty, so maintaining the same span length across different levels does not make sense.

If a user succeeds at a task some number M times consecutively, the sequence length will increase by one. Similarly, if the user fails at a task some number N times consecutively, the sequence length will decrease by one. The spans have lower and upper bounds to avoid trivial levels as well as the upper limits of working memory.

The dynamic difficulty system of *Caribbean Quest* helps address one of the largest challenges with *Cognitive Carnival*: finding an appropriate difficulty level for all users. Although this initial attempt is coarse-grained, and the difficulty increases/decreases in large amounts, it is still preferable to completely fixed span lengths. By operating per-user, the dynamic difficulty system will self-correct as necessary, and should–in a broad manner–account for the intrinsic ability of each participant. This addresses Objective 1.3.

4.6 Games of Caribbean Quest

4.6.1 Scuba

In the *Scuba* game, the player navigates a scuba diver around a coral reef, collecting items and avoiding enemy fish. The *Scuba* game is analogous in the nature of the cognitive tasks to the *Platform* game from *Cognitive Carnival*, and a similar side-view of a game environment containing an avatar navigated by the player. At the beginning of each round, the player is given a sequence of tokens to collect; the player must move around the reef to collect the items in the correct sequence. The reef is seen from the side, with the coral forming a maze-like structure. Levels may contain hazards such as puffer fish that move around the maze and stun the player temporarily if they are touched.

Scuba improves upon the *Platform* game from *Cognitive Carnival* primarily from an easier to use control scheme. It was observed in the *Cognitive Carnival* interventions that although participants tended to enjoy the *Platform* game, several participants found it the most difficult to play. Platforming games in general tend to require

skill and dexterity to time jumps and maintain fine control of a player character on a narrow structure or ladder. I wished to eliminate these fine motor control requirements; the coral reef maze structure allows for much more forgiving controls while still giving the participant freedom of movement and a level to explore and discarding some of the discouraging and overly-difficult game elements.

Figure 4.5: The *Scuba* game from *Caribbean Quest*. In this level, the tokens to collect are garbage that is littering the coral reef. The colour of the tokens is fixed (blue), but the image varies (boat, gas can, oil drum). This level has enemies enabled; the puffer fish patrol around the reef in a pattern and stun the player if contacted.

4.6.2 Pirate Delicatessen

In *Pirate Delicatessen*, the player must recollect a sandwich recipe by selecting the ingredients in the correct order. This game is analogous to *Liftoff* from *Cognitive Carnival* in both gameplay and cognitive elements. *Pirate Delicatessen* incorporated all of the elements present in *Liftoff*, but provided more colourful and thematic objects for the participant to interact with. The game also uses the *Caribbean Quest* working memory dynamic difficulty system, allowing for more appropriate difficulty level for

each participant.

Pirate Delicatessen expands *Liftoff* by providing more parameters to use (e.g. more sophisticated movement of objects, which can collide with each other and rotate using a physics simulation). The full *Pirate Delicatessen* parameters are listed in Table A.5.

Figure 4.6: The *Pirate Delicatessen* game from *Caribbean Quest*. In this level, the objects to remember are hamburger ingredients (tomato, onion, bun).

4.7 Sustained Attention Games

The sustained attention games, *Submarine* and *Wave*, require the user to focus for periods on the order of two to six minutes, performing a small task at regular intervals. The games are similar in cognitive tasks, and both contain visual and auditory distractors. They differ entirely in gameplay, however: *Submarine* is a static environment where the participant's only control is selecting a fish from the middle porthole, whereas *Wave* allows the user to navigate an avatar in an ocean environment.

The sustained attention games include a miniature sailboat in the progress bar which gradually moves left to right to indicate the time remaining in the level. Al-

though this is redundant, as there is a timer, it is intended to provide a more satisfying visual representation than that of a clock.

4.7.1 Submarine

In the *Submarine* game, the player observes fish out of five portholes of a submarine. They must press an action button when the correct fish passes through the centre porthole. *Submarine* is analogous to *Wheel* from *Cognitive Carnival* in both gameplay concept and cognitive tasks, but disposes of the somewhat problematic 3D elements.

Figure 4.7: The *Submarine* game from *Caribbean Quest*. This is an early level, so none of the portholes are blocked. When the blue fish passes through the centre porthole, the user must select it by pressing a button. The fish move at a relatively slow pace; on later levels their speed increases, requiring the participant to react more quickly.

The submarine setting provides a great variety of distractor elements. Lights flash on and off, glowing lines scroll on TV screens, and various marine sounds will play at random intervals. The *Submarine* parameters are listed in Table A.6.

4.7.2 Wave

The *Wave* game is a continuous performance game similar to *Submarine* for cognitive tasks. However, in *Wave*, the player controls a ship which they can manoeuvre on screen to collect targets, providing for a more interactive game experience. *Wave* contains a variety of distracting elements, such as birds that will fly across the screen. The game also has obstacles, such as rocks, that must be avoided. The *Wave* parameters are detailed in Table A.7.

Figure 4.8: The *Wave* game from *Caribbean Quest*. This two minute level tasks the participant to navigate their submarine avatar to collect specific garbage items from the ocean.

4.8 Selective Attention Games

Selective attention tasks require the participant to identify the presence or absence of visual elements on the screen by selectively attending to specific details while quickly scanning the visual array of objects. This can be confounded by several factors, such as the similarity of the object to its surroundings.

4.8.1 Squidditch

Squidditch is an entirely new type of game from those in *Cognitive Carnival*. In *Squidditch*, the player is shown a target object to remember. Then, a large array of other objects are displayed on the screen that appear similar to the object but differ in at least one property (e.g. different colour). The target fish may or may not be present on the screen. The participant must visually scan selectively attending to all elements in the array to determine if the target object is present or absent. The objects used in *Squiddtich* are mostly various (cartooned) types of marine life, like starfish and squid.

Figure 4.9: The *Squidditch* game from *Caribbean Quest*. The target object is a yellow starfish with five points, on the middle-left of the screen. The remaining objects are a mix of yellow starfish with six points, and purple starfish with five points.

As the participant progresses to more difficult levels, the distinction between objects is narrowed by using assets that are much more similar in shape and colour. This provides a gradual increase in the cognitive load. See Table A.10 for a full listing of selective attention parameters.

4.9 Motivational Structure

4.9.1 Bonus games and the Sand Dollar currency

I wished to add as much intrinsic motivation as possible in *Caribbean Quest*. Even the small amount of intrinsic rewards in *Cognitive Carnival* were observed to aid with motivation (Section 3.4.1). Motivation is also related to the duration of materials: finishing the game exhausted all sources of reward in *Cognitive Carnival*, causing some participants to have a massive decrease in interest.

In order to address this problem, I separated trophies from fixed objectives (e.g. "Complete Level 4") and instead introduced an intermediary in-game currency, the "Sand Dollar", to facilitate this. When participants successfully complete some amount of cognitive training, they are taken to a bonus game where they have the opportunity to earn Sand Dollars. Afterwards, these can be spent at the store to purchase trophies (see Section 4.9.3).

Figure 4.10: The *Scuba* bonus game from *Caribbean Quest*. The usual *Scuba* objects, such as garbage, are replaced with Sand Dollars. The Sand Dollars fall from above and roll around the coral reef. The player has a time limit to collect as many as possible.

Figure 4.11: The *Submarine* bonus game from *Caribbean Quest*. Instead of fish passing by the submarine left-to-right, Sand Dollars and puffer fish move rapidly downwards. The player has three buttons which correspond to the three open portholes and select the object passing through. When a Sand Dollar appears and the player selects it, it is awarded to the player. However, when the player selects a puffer fish, an alarm sounds, lights flash, and the user is "locked out", i.e. prevented from collecting any further Sand Dollars, for a few seconds. This means that to earn the maximum number of possible Sand Dollars, the player can't simply mash the three buttons repeatedly.

Figure 4.12: The *Wave* bonus game from *Caribbean Quest*. Sand Dollars appear at the top of the screen, and the player must quickly navigate the submarine avatar to collect them before they pass the bottom of the screen.

The bonus games recycled assets from the existing games to save on development time. I created bonus versions of *Scuba*, *Submarine*, and *Wave* by replacing the cognitive sequence tokens with Sand Dollars. I decided that the *Squidditch* and *Pirate Delicatessen* games did not lend themselves well to bonus versions, so I created an entirely new bonus game, *Cannon*, to serve as the bonus game for them. In the *Cannon* game, the player must take aim at coins with a cannon and try to collect as many as possible.

Figure 4.13: The *Cannon* bonus game from *Caribbean Quest*. This is the bonus game for *Squidditch* and *Pirate Delicatessen*. The participant aims the cannon and shoots the cannon ball, trying to touch as many Sand Dollars as possible with the cannon ball, or the crates and barrels. When any of these collide with the Sand Dollars, they are awarded to the participant. This bonus game relies heavily on the Farseer Physics Engine for Silverlight to calculate the movement of the cannonball, crates, and barrels.

4.9.2 The Overworld

A game *Overworld* is an overloaded term, but typically refers to an abstract high-level view of a game, indicating objectives (vertices) and paths to objectives (edges). This graph structure can be used as a dependency graph, to enforce partial or full order of the set of objectives, as in *Super Mario Bros 3.* (1991) [36] and the home version of *Bionic Commando* [12] (1988). I decided to implement an Overworld for *Caribbean Quest* to provide a visual of the progress for each game, and to combine all of the games onto one overall map. This creates a sense of unity between the otherwise independent games.

Each node on the Overworld is a coloured dot. Red dots indicate incomplete

levels that you can't play yet. Yellow dots indicate incomplete levels that can be attempted. Green dots indicate completed levels. As the participant successfully completes objectives, the path lights up green, until the end is reached and a large reward and "congratulations" screen is presented. The dots are then reset. This idea was borrowed from the game *Treasure Mountain!* (1990) [48], so that the game is meant to be completed repeatedly, each time providing a token of success. This avoids the problem of trying to provide enough content for all users while also providing concrete long-term goals that transcend a single typical 30-minute session.

Figure 4.14: Part of the Overworld from *Caribbean Quest*, showing the levels for the *Wave* game. The first level (middle-left portion of the screen) has been completed and is coloured green. The next level is yellow as it is available to play. The remainder are red and inaccessible.

4.9.3 Store

Caribbean Quest's bonus games reward the player with the Sand Dollar in-game currency. These can be spent in the store to purchase various reward items. Once an item is purchased, it is displayed on the Overworld. This combines the reward

progress with the overall progress into one visual system. I believe that this will aid in motivation to earn Sand Dollars, in turn increasing the motivation to complete cognitive tasks.

Figure 4.15: The same part of the Overworld from Figure 4.14. The user has completed another level, and the dots are updated to reflect his progress. The user has spent 1000 Sand Dollars at the store on a reward item: a sandcastle. This appears on the lower middle portion of the screen.

Figure 4.16: The Store from *Caribbean Quest*. The rewards are displayed in a grid of icons which can be selected for purchase using Sand Dollars.

4.10 Summary

Caribbean Quest expanded the *Cognitive Carnival* suite of games. Cognitive and game parameters were separated into distinct components, allowing games to re-use cognitive parameter classes. Furthermore, this separation allowed for a uniform interface to access working memory and sustained attention sequences. The working memory sequence generation was improved, resulting in a robust backtracking algorithm with guaranteed success or failure, as well as the ability to add new objects and attributes in the future. Finally, a dynamic difficulty system for working memory was implemented, which responds to the user's current performance and adjusts the span length accordingly.

Chapter 5

Conclusions and Future Work

5.1 Conclusions

5.1.1 *Cognitive Carnival*

It is important to emphasize that the games were not provided in a vacuum. While it is tempting to claim that the games intrinsically improved sustained attention and working memory, evidenced by the changes in the pre- and post- test psychological measures from field studies using *Cognitive Carnival*, the games were a material used with trained psychology research interventionists, who provided children with metacognitive strategies and support through the games. It is not only incorrect to assume that the game play alone can result in improvement, but dangerous: misunderstanding the role that the games play as part of a larger support system and intervention can undermine the entire process for children who struggle with attention and memory deficits.

Nevertheless, *Cognitive Carnival* demonstrated as a proof-of-concept that working memory, sustained attention, and selective attention can be integrated into games, satisfying Objective 1.1 (*Integrate working memory, sustained attention, and selective attention training material into video games*). To achieve this, Objective 1.2 (*Compile the working memory, sustained attention, and selected attention training options that control the cognitive difficulty into collections of concrete data types, or "parameters".*) was attained via the parameters listed in Appendix A.1.

5.1.2 Caribbean Quest

Motivation is a crucial element in interventions, as children especially can find the arduous tasks boring and tedious, and may refuse outright to participate. Engaging the children in the exercises is therefore of the utmost importance.

The feedback from interventionists and participants in the studies of Chapter 3 indicated that dynamic difficulty for working memory was necessary to allow the working memory games to adjust to each participant's ability. I implemented a dynamic difficulty system which modifies the span length to set an appropriate difficulty level, on a per-level, per-user basis. In future studies, this should help reduce frustration with overly difficult tasks, and boredom with overly simple tasks, while maintaining a challenge for each user. This satisfies Objective 1.3 (*Design a software component to dynamically adjust the difficulty of the working memory games to accommodate each participant's current performance*).

The games of *Cognitive Carnival*, in hindsight, had intersecting cognitive difficulty parameters. Furthermore, each game had distinct game difficulty parameters. This led to Objective 1.4 (Determine the software relationship between the games and the cognitive components of difficulty). In *Caribbean Quest*, such a separation was determined, and separate cognitive difficulty and game difficulty parameter classes were constructed, satisfying the objective. These are listed in Appendix A.2.

One of the most open challenges of the games was expressed in Objective 1.5 (*Find a way to make the amount of materials adjustable and keep motivation high, while still having distinct goals and rewards*). Not having this design caused decreases in motivation in the *Cognitive Carnival* studies. Therefore, in *Caribbean Quest*, I endeavoured to find a way for the games to be more flexible in materials. To accomplish this, I designed the games to give a currency instead of a set list of rewards. By using the currency, each level provides motivation even if it has already been completed. Furthermore, there are still distinct long-term objectives of completing each game's entire set of levels.

Objective 1.6 (*Allow a flexible and expressive way to describe the game parameters, independently of the game binary*) required a way to describe the cognitive parameters outside of the code. I accomplished this using XML schemata to specify each level's characteristics. The XML files can be modified outside of the game binary and override the game's default content, allowing interventionists to modify the therapy at will.

Finally, I determined from the complexity of the working memory parameters, that the working memory sequence generator needed to be improved to provide for a stable and predictable program, outlined in Objective 1.7 (*Create a robust algorithm for generating working memory strings, and a uniform interface to access it*). Section 4.4.2 details the design and implementation of such an algorithm. Furthermore, the algorithm accepts the working memory parameters, instead of a particular game's parameters, providing for a uniform interface.

5.1.3 Final Remarks

Cognitive Carnival was a collaborative experiment developed iteratively via trial-and-error. There was no professional art. Sounds were recorded and edited by amateurs, and I used stock graphics liberally. Despite this, at least one participant in the study at the end asked a research assistant where he could buy *Cognitive Carnival*. I am confident that *Caribbean Quest* will–in addition to its primary goals of web delivery, content scaling, and dynamic difficulty–increase engagement by including professionally produced music, sound, and art; improved motivational structure; and more rewarding gameplay.

5.2 Future Work

The development and evaluation of the game suites revealed some future computer science research paths that have the potential to be instrumental in making cognitive games more effective.

5.2.1 Parameters

The parameters of the games have many valid combinations, but many invalid ones as well. It would be useful to have a way of reducing the possibility of the program parsing incorrect information. One method could be to use more specific XML schemata to limit not only the data types allowed, but ranges of data (e.g. sequence lengths should always be positive integers). More sophisticated improvements could be added by using rules for parameters to restrict the allowed combinations. For example, the "Blackout" parameter used in working memory games only makes sense if the "Auditory" parameter is enabled (otherwise, there's no way of learning what

the sequence is). By explicitly indicating this requirement, vast numbers of invalid parameters could be culled.

5.2.2 Progress

It would be useful to provide more structured control over the overall progress of the games. The games required the interventionist to determine the order of the games and time spent per game, for each user session. Keeping the games open was necessary to provide flexibility and to allow the participants to decide the order of the games. Expanding the Overworld concept to use a more sophisticated dependency graph to enforce time constraints and game orders would allow the game to guide the user through a particular regime.

5.2.3 Measuring and controlling difficulty

In the opinion of the author, the difficulty aspects of the program is the most relevant to computer science. Developing better models of difficulty and using these to allow for more gradual progression through the games, on a per-user basis, is a core design challenge.

Working memory difficulty

The establishment of difficulty metrics for the many parameters in working memory spans would be of great benefit to developing working memory games. Without knowing what makes a working memory task difficult, it is perilous to concoct an algorithm to navigate the user through the many-dimensional parameter space of the working memory tasks. Only the low-hanging fruit of changing the span length and the amount of distracting elements currently provides an unambiguous method of adjusting the difficulty.

Metrics could provide the foundation for a truly dynamic difficulty system and fine-grained control, reducing user frustration and maximizing their performance by placing them at the most appropriate level on an individual basis. This affects not only the enjoyment of the task, but the results of the task itself.

Interaction of game difficulty and cognitive difficulty

In addition to understanding and modelling the difficulty of the cognitive tasks, it would also be of great benefit to know how specific game types affect user performance. Since the games in *Caribbean Quest* have distinct cognitive and game parameters, this design could lend itself to studying performance on identical cognitive tasks across different games.

Bibliography

[1] Adobe Flash. http://www.adobe.com/products/flash.html. Accessed: 18/02/2012.

[2] Microsoft .NET Framework. http://www.microsoft.com/net. Accessed: 18/02/2012.

[3] Microsoft Silverlight. http://www.microsoft.com/silverlight/. Accessed: 18/02/2012.

[4] Microsoft XNA. http://create.msdn.com/en-US/. Accessed: 18/02/2012.

[5] The C Sharp Programming Language. msdn.microsoft.com/en-us/vstudio/hh388566. Accessed: 18/02/2012.

[6] XBox 360. http://www.xbox.com/en-US/. Accessed: 18/02/2012.

[7] D. Bartle, S. Rossoff, D. Whittaker, B. Gooch, K. Kerns, and J. MacSween. Cognitive games as therapy for children with FAS. In *ACM SIGGRAPH 2010 Posters*, page 48. ACM, 2010.

[8] BioWare. Mass Effect. XBox 360, PlayStation 3, Microsoft Windows, 2007.

[9] G. Blennow, J. Heijbel, P. Sandstedt, and B. Tonnby. Discontinuation of antiepileptic drugs in children who have outgrown epilepsy: Effects on cognitive function. *Epilepsia*, 31:S50–S53, 1990.

[10] Milton Bradley. Simon, 1978.

[11] T. Bray, J. Paoli, C.M. Sperberg-McQueen, E. Maler, F. Yergeau, et al. Extensible markup language (XML) 1.0, 2000.

[12] Capcom. Bionic Commando. Nintendo Entertainment System, 1988.

[13] A.E. Chudley, J. Conry, J.L. Cook, C. Loock, T. Rosales, and N. LeBlanc. Fetal alcohol spectrum disorder: Canadian guidelines for diagnosis. *Canadian Medical Association Journal*, 172(5 suppl):S1, 2005.

[14] S. Cooper, F. Khatib, A. Treuille, J. Barbero, J. Lee, M. Beenen, A. Leaver-Fay, D. Baker, Z. Popovic, et al. Predicting protein structures with a multiplayer online game. *Nature*, 466(7307):756–760, 2010.

[15] A. Diamond and K. Lee. Interventions shown to aid executive function development in children 4 to 12 years old. *Science*, 333(6045):959–964, 2011.

[16] Douglas E. Smith. Lode Runner. Apple][, VIC-20, Commodore 64, PC Booter, ZX Spectrum, Atari XL/XE, SG-1000, XBLA, Microsoft Windows, iPod, Macintosh, Virtual Console, PlayStation Network, BBC Micro, Atari Lynx, PlayStation, Nintendo Entertainment System, Super NES, Amstrad CPC, 1983.

[17] S. Egenfeldt-Nielsen. Beyond Edutainment: Exploring the Educational Potential of Computer Games. *future*, 2005.

[18] Erin Catto. Box2D. C++ source code (zlib license), 2004.

[19] P.S. Fastenau, J. Shen, D.W. Dunn, and J.K. Austin. Academic underachievement among children with epilepsy. *Journal of learning disabilities*, 41(3):195–207, 2008.

[20] Armor Games. Crush the Castle. Microsoft Windows, Mac OS X, iOS, Android, 2009.

[21] Epic Games. Unreal Engine. Microsoft Windows, Mac OS X, PlayStation 3, iOS, 1998.

[22] J. Garcia-Barcena and A. Garcia-Crespo. Game based learning: a research on learning content management systems. *interaction*, 9(11):12, 2006.

[23] J.E. Gardner. Can the Mario Bros. help? Nintendo games as an adjunct in psychotherapy with children. *Psychotherapy: Theory, Research, Practice, Training*, 28(4):667, 1991.

[24] J.G. Hogle. Considering games as cognitive tools: In search of effective Edutainment. *University of Georgia Department of Instructional Technology*, 1996.

[25] J. Holmes, S.E. Gathercole, D.L. Dunning, K.A. Hilton, J.G. Elliott, et al. Working memory deficits can be overcome: Impacts of training and medication on working memory in children with ADHD. *Applied Cognitive Psychology*, 24(6):827–836, 2010.

[26] R. Hunicke. The case for dynamic difficulty adjustment in games. In *Proceedings of the 2005 ACM SIGCHI International Conference on Advances in computer entertainment technology*, pages 429–433. ACM, 2005.

[27] R. Hunicke and V. Chapman. Ai for dynamic difficulty adjustment in games. In *Challenges in Game Artificial Intelligence AAAI Workshop*, pages 91–96, 2004.

[28] W. James, F. Burkhardt, and I.K. Skrupskelis. *The principles of psychology*, volume 1. Harvard Univ Pr, 1981.

[29] W.O. Kalberg, B. Provost, S.J. Tollison, B.G. Tabachnick, L.K. Robinson, H. Eugene Hoyme, P.M. Trujillo, D. Buckley, A.S. Aragon, and P.A. May. Comparison of motor delays in young children with fetal alcohol syndrome to those with prenatal alcohol exposure and with no prenatal alcohol exposure. *Alcoholism: Clinical and Experimental Research*, 30(12):2037–2045, 2006.

[30] K.A. Kerns, J. MacSween, S. Vander Wekken, and V. Gruppuso. Investigating the efficacy of an attention training programme in children with foetal alcohol spectrum disorder. *Developmental Neurorehabilitation*, (0):1–10, 2010.

[31] W.R. Livermore. *The American Kriegsspiel. A game for practicing the art of war upon a topographical map.* Houghton, Mifflin and Co., 1879.

[32] A.R. Luria. Restoration of function after brain injury. 1963.

[33] J. MacSween, S. Vander Wekken, K. Sinclair, D. Bartle, and K. Kerns. Investigating the efficacy of an computerized attention training program in children with epilepsy. Poster session presented at the 39th Annual Meeting of the International Neuropsychological Society, Boston, Massachusetts, USA., 2011.

[34] P.A. May, J.P. Gossage, W.O. Kalberg, L.K. Robinson, D. Buckley, M. Manning, and H.E. Hoyme. Prevalence and epidemiologic characteristics of FASD

from various research methods with an emphasis on recent in-school studies. *Developmental disabilities research reviews*, 15(3):176–192, 2009.

[35] D. Nieborg. America's Army: More than a game, 2004.

[36] Nintendo. Super Mario Bros. 3. Family Computer, 1988.

[37] Press Play. Max and the Magic Marker. Microsoft Windows, Windows Phone 7, Nintendo DS, PlayStation Network, Wii WiiWare, Mac OS X, XBox, XBox 360, Linux, 2010.

[38] P.J.M. Prins, S. Dovis, A. Ponsioen, E. Ten Brink, and S. Van der Oord. Does computerized working memory training with game elements enhance motivation and training efficacy in children with ADHD? *Cyberpsychology, Behavior, and Social Networking*, 14(3):115–122, 2011.

[39] Petri Purho. Crayon Physics Deluxe. Microsoft Windows, Mac OS X, iOS, Linux, 2009.

[40] C. Rasmussen. Executive functioning and working memory in fetal alcohol spectrum disorder. *Alcoholism: Clinical and Experimental Research*, 29(8):1359–1367, 2005.

[41] A. Rizzo, B.O. Rothbaum, and K. Graap. Virtual reality applications for the treatment of combat-related PTSD. 2007.

[42] Valve Software. Half-Life 2. Microsoft Windows, Mac OS X, XBox, XBox 360, 2004.

[43] Valve Software. Source. Microsoft Windows, Mac OS X, XBox, XBox 360, 2004.

[44] T. Stach, TC Graham, J. Yim, and R.E. Rhodes. Heart rate control of exercise video games. In *Proceedings of Graphics Interface 2009*, pages 125–132. Canadian Information Processing Society, 2009.

[45] Pandemic Studios. Full Spectrum Warrior. DVD-DL,CD-ROM, 2004.

[46] K.B. Suzman, R.D. Morris, M.K. Morris, and M.A. Milan. Cognitive-behavioral remediation of problem solving deficits in children with acquired brain injury. *Journal of behavior therapy and experimental psychiatry*, 28(3):203–212, 1997.

[47] M. Tan and R. Appleton. Attention deficit and hyperactivity disorder, methylphenidate, and epilepsy. *Archives of disease in childhood*, 90(1):57, 2005.

[48] The Learning Company. Treasure Mountain! MS-DOS, 1990.

[49] H.S. Thompson. XML Schema Part 1: Structures Second Edition, 2004.

[50] B. van Twillert, M. Bremer, and A.W. Faber. Computer-generated virtual reality to control pain and anxiety in pediatric and adult burn patients during wound dressing changes. *Journal of Burn Care & Research*, 28(5):694, 2007.

[51] Jeff Weber. Farseer Physics Engine. C# source code (Ms-PL license); Microsoft XNA, Silverlight, 2006.

[52] C. Yun, P. Trevino, W. Holtkamp, and Z. Deng. PADS: Enhancing gaming experience using a profile-based adaptive difficulty system. In *Proceedings of the 5th ACM SIGGRAPH Symposium on Video Games*, pages 31–36. ACM, 2010.

[53] Zimmermann, P. and Gondan, M. and Fimm, B. Kitap: Testbatteriezur aufmerksamkeitspru fung fur kinder [test battery for attention assessment in children], 2002.

Appendix A

Parameters and Source Code Listings

The data types listed in these parameters are part of the Microsoft .NET Framework. **Int32** is a signed 32-bit integer; **Enum** is an enumerated type; **Single** and **Double** refer to single-precision and double-precision floating point values, respectively; **TimeSpan** is a quantity of time. The entire range of values of a type are not necessarily valid arguments.

A.1 Parameters from *Cognitive Carnival*

Parameter	**Type**	**Description**
Length	Int32	The length of the sequence.
Secondary	Boolean	Requires the user to select a target *if* it appears after some particular target.
Distractors	Int32	Additional n distractors displayed outside of the wheel.
Distractor Distance	Double	Proximity of distractors to the centre of the screen, in pixels.
Speed	Double	The number of seconds each target is available to select.

Table A.1: *Wheel* parameters.

Parameter	**Type**	**Description**
Length	Int32	The length of the sequence.
Level	Int32	The number of the particular environment set (arrangement of platforms, ladders, obstacles, targets, etc.).
Distractors	Int32	n tokens that are not part of the sequence.
ItemType	Enum	The type of item to be collected (e.g. "Sports Balls").
Nameable	Boolean	Whether or not the item type has an easily identifiable name (e.g. "Fruits" are nameable, but abstract shapes are not).
Auditory	Boolean	Presents the sequence via auditory instructions. Requires the sequence to be Nameable.
Reverse	Boolean	Presents the order of the sequence in reverse.
Ability	Int32	A per-participant parameter set by the interventionist. This offset is added to the *Length*. (e.g. a participant with a higher-than-average ability may have an offset of two. This means all sequences have an additional two items. This parameter was added after the Pilot study (Section 3.2)).

Table A.2: *Platform* parameters.

Parameter	**Type**	**Description**
Length	Int32	The length of the sequence.
Order	Boolean	Requires the user to repeat the sequence in numerical or alphabetical order (depending on context).
Reverse	Boolean	Requires the user to repeat the sequence in reverse.
Distractors	Boolean	Additional n distractors after the sequence is displayed that the user must ignore, where $n = Length$.
Flash	Boolean	The sequence tokens flash and then disappear, only reappearing for selection at the end of the animation.
Blackout	Boolean	The sequence only appears after the presentation (requires auditory).
Speed	Double	The amount of time each token is animated in seconds.
Sound	Boolean	Plays a specific sound for each token type when animating.
Line	Boolean	Displays the sequence tokens randomly in a line after animation, instead of their original positions.
Ignorable	Int32	Instructs the user to ignore n tokens with specific properties (shape, colour) in the sequence.

Table A.3: *Liftoff* parameters.

A.2 Parameters from *Caribbean Quest*

A.2.1 Game parameters

Parameter	Type	Description
EnemiesEnabled	Boolean	Adds or removes enemy fish from the game to increase/decrease game difficulty.
EnemySpeed	Double	The speed of the enemies, in pixels per second.
Environment	Int32	Indicates which of the environments (backdrop, geometry, etc.) to use.

Table A.4: *Scuba* Parameters.

Parameter	Type	Description
Animation	Boolean	Whether or not the sequence is animated.
Flash	Boolean	The objects will only be visible when they are presented.
Blackout	Boolean	Do not display the sequence until the instructions have finished (requires auditory).
Layout	Enum	Preset layouts for the objects (e.g. "Spread Out", "Linear").
InitialLayout	Layout	The layout of the objects before the animation begins.
FinalLayout	Layout	The layout of the objects when the animation is finished.
Movement	Boolean	When the sequence has completed animating, the objects move around the screen randomly.
Rotation	Boolean	If movement is enabled, also allows the objects to rotate.
AnimationDuration	Double	The amount of time to display each item when it is animated.

Table A.5: *Pirate Delicatessen* parameters.

Parameter	**Type**	**Description**
BlockOut	String	Indicates which of the 5 portholes are obscured.
Distractors	Boolean	Animates various elements of the submarine and plays random marine sounds.

Table A.6: *Submarine* parameters.

Parameter	**Type**	**Description**
Distractors	Boolean	Adds elements like seagulls flying across the screen and random marine sounds.
Obstacles	Boolean	Adds obstacles such as rocks to stun the player's ship.

Table A.7: *Wave* parameters.

A.2.2 Cognitive parameters

Parameter	Type	Description
Length	Int32	The number of tokens in the WM string, including "ignorables", but excluding "extras".
TextType		Requires the user to select a target *if* it appears after some particular target.
Extras	Int32	Additional n distractors displayed outside of the wheel.
Distractors	Int32	The number of tokens in the WM string that are to be ignored when repeated.
DistinctColours	Boolean	Each colour must be distinct for all tokens.
DistinctCharacters	Boolean	Each character must be distinct for all tokens.
DistinctShapes	Boolean	Each shape must be distinct for all tokens.
OverallDistinct	Boolean	Each token must be distinguishable from the others.
FixedShape	Boolean	All tokens are the same geometric shape or image.
FixedColour	Boolean	All tokens are the same colour.
FixedCharacter	Boolean	All tokens have the same character.
IgnorableColours	Int32	The number of ignorable colours in the working memory string.
IgnorableShapes	Int32	The number of ignorable shapes in the working memory string.
IgnorableCharacters	Int32	The number of ignorable characters in the working memory string.
Auditory	Boolean	The sequence instructions are auditory.
RedundantAuditory	Boolean	The auditory instructions will describe all aspects of each token, even if the aspects are redundant (e.g. all tokens are red but have distinct shapes, the auditory instructions will include the word "red" during the presentation.
ImageCategory	Enum	The category of image (Garbage, Sandwich Ingredients, Shapes, etc.).

Table A.8: Working memory parameters.

Parameter	Type	Description
Duration	TimeSpan	The duration of the trial.
Percent	Single	The minimum percentage of correct targets required to pass.
NBack	Int32	The distance in the NBack task.
Matching	Boolean	Match symbols with others ("if you see target X, then target Y following it is valid").
Speed	Double	The rate at which the sequence scrolls.
Type	Enum	The type of targets (e.g. "Fish").

Table A.9: Continuous performance parameters.

Parameter	Type	Description
NumOfDisplayTargets	Int32	The number of targets to display on the screen.
NumOfTrials	Int32	The number of trials in each level.
TrialDuration	TimeSpan	The maximum amount of time allowed to complete each trial.
GridSize	Double	The size of the object grid in pixels.
ObjectSize	Double	The scale of the objects, where 1.0=64 square pixels.
Movement	Boolean	The objects move while the participant is scanning.
TargetType	Enum	The type of targets (e.g. "Starfish").
DisplayCorrect	Boolean	Show the user the target fish (if present) after his or her response.
BackgroundImage	Enum	The backdrop behind the objects.

Table A.10: Visual scanning parameters

A.2.3 Backtracking algorithm for working memory sequence generation

```
private void backtrack(int depth, int sequenceIndex, List<Attribute>
   backtrackAttributes)
{
            //Leaf node reached
            if (depth == -1)
            {
                return;
            }
            else
            {
                Attribute currentAttribute = backtrackAttributes[
                   backtrackAttributes.Count - depth - 1];

                //Random, mod length
                int length = currentAttribute.Total;

                int randomIndex = random.Next(length);
                bool succeeded = false;

                for (int i = 0; i < length; i++)
                {
                    //Is this attribute preselected and not random?
                    if (currentAttribute.Preselected)
                    {
                        indices[sequenceIndex][backtrackAttributes.Count
                            - depth - 1] = currentAttribute.
                           NonRandomIndex;
                        succeeded = true;
                        break;
                    }
                    //Is this attribute unique? Or can we ignore the
                       used array and select duplicates?
                    else if (currentAttribute.Unique)
                    {
                        if (!currentAttribute.Used[randomIndex])
                        {
                            currentAttribute.Used[randomIndex] = true;
                            indices[sequenceIndex][backtrackAttributes.
                               Count - depth - 1] = randomIndex;
```

```
                succeeded = true;
                break;
            }
            else
            {
                //increment modulo length
                randomIndex = (randomIndex + length + 1) %
                    length;
            }
        }
        //Not unique, pick anything (except ignorables)
        else
        {
            //Make sure we're not creating an ignorable
            if (currentAttribute.IgnorableCount > 0)
            {
                if (currentAttribute.IgnorableIndex ==
                    randomIndex)
                {
                    //increment modulo length
                    randomIndex = (randomIndex + length + 1)
                        % length;
                    continue;
                }
            }

            indices[sequenceIndex][backtrackAttributes.Count
                - depth - 1] = randomIndex;
            currentAttribute.Used[randomIndex] = true;
            succeeded = true;
            break;
        }
    }

    //Predicate satisfied
    if (succeeded)
    {
        backtrack(depth - 1, sequenceIndex,
            backtrackAttributes);
    }
    else
    {
```

```
                    //Failed, stop backtracking.
                    backtrackFailed = true;
                    throw new ImpossibleSequenceException("Backtracking␣
                        failed␣at␣depth␣" + depth + "␣on␣attribute␣" +
                        currentAttribute);
                }
            }
}
```