

8-14-2016

Code Girl

Amanda Holl
Santa Clara University

Follow this and additional works at: http://scholarcommons.scu.edu/cseng_mstr



Part of the [Computer Engineering Commons](#)

Recommended Citation

Holl, Amanda, "Code Girl" (2016). *Computer Science and Engineering Master's Theses*. Paper 1.

This Thesis is brought to you for free and open access by the Engineering Master's Theses at Scholar Commons. It has been accepted for inclusion in Computer Science and Engineering Master's Theses by an authorized administrator of Scholar Commons. For more information, please contact rscroggin@scu.edu.

SANTA CLARA UNIVERSITY
DEPARTMENT OF COMPUTER ENGINEERING

Date: August 14, 2016

I HEREBY RECOMMEND THAT THE THESIS PREPARED UNDER MY SUPERVISION BY

Amanda Holl

ENTITLED

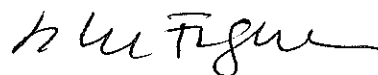
Code Girl

BE ACCEPTED IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE DEGREE OF

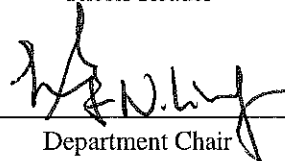
MASTER OF SCIENCE IN COMPUTER SCIENCE AND ENGINEERING



Thesis Advisor



Thesis Reader



Department Chair

Code Girl

by

Amanda Holl

Submitted in partial fulfillment of the requirements
for the degree of
Master of Science in Computer Science and Engineering
School of Engineering
Santa Clara University

Santa Clara, California
September 8, 2016

Code Girl

Amanda Holl

Department of Computer Engineering
Santa Clara University
September 8, 2016

ABSTRACT

Despite the growing importance of technology and computing, fewer than one percent of women in college today choose to major in computer science. Educational programs and games created to interest girls in computing, such as *Girls Who Code* and *Made With Code*, have been successful in engaging girls with interactive and creative learning environments, but they are too advanced for young girls to benefit from. To address the lack of educational, computer science games designed specifically for young girls, we developed a web-based application called *Code Girl* for girls age five to eight to customize their own avatars using Blockly, an open-source visual coding editor developed by Google. Girls learn basic computer science and problem-solving skills by successfully using puzzle-piece like blocks to complete challenges that unlock new accessories for their avatars.

In conducting user testing, we assessed the usability and complexity of the application and identified ways to better meet research goals of educating and inspiring young girls to pursue computer science. The overall feedback we received on *Code Girl* in user testing was positive, as a majority of the girls expressed an interest in playing the game again and playing more games designed to teach programming. *Code Girl* thus appeals to the general pastimes of young girls to interest them in computer science from an early age and hopefully inspires them to pursue computing as a career. The application, which though it can continue to be developed with even more challenges and clothes and accessories, and thereby teach additional concepts, is ready to be released to the public. Initial steps are provided to incorporate *Code Girl* into educational programs at local and national levels.

Table of Contents

1	Introduction	1
1.1	Problem Statement	1
1.2	Background	2
1.3	Requirements	3
1.3.1	Functional Requirements	4
1.3.2	Non-functional Requirements	4
1.3.3	Design Constraints	5
1.4	Conclusions	5
2	Design	6
2.1	Conceptual Model	6
2.1.1	State Diagram	6
2.1.2	Activity Diagrams	7
2.1.3	Mockups	9
2.2	Use Cases	11
2.3	Architectural Design	14
2.3.1	Avatar Customization	14
2.3.2	Challenges	14
2.3.3	Game Logic	14
2.3.4	Loading Logic	15
2.3.5	Database	15
2.4	Technologies Used	15
2.5	Design Rationale	17
2.5.1	Game Design	17
2.5.2	Visual Programming Environment	18
2.5.3	Blockly	18
2.5.4	Technologies	19
2.5.5	Responsive Design	19
2.5.6	Story-telling	20
3	Application Overview	23
3.1	Create and Account/Login	24
3.2	Avatar Customization	24
3.3	Challenges	26
3.3.1	Challenge 1: Introduction to Blockly	26
3.3.2	Challenge 2: Sequentiality	27
3.3.3	Challenge 3: Loops	28
3.3.4	Challenge 4: If-Else Conditional Statement	29
3.3.5	Challenge 5: Do-Until Conditional Statement	29
3.3.6	Challenge 6: Nested Conditional Statements	30
3.3.7	Challenge 7: Order and Conditions	31
3.4	Saving the Avatar	32

4	Project Management	34
4.1	Test Plan	34
4.2	Project Risks	34
4.3	Societal Issues	36
4.3.1	Ethics	36
4.3.2	Social Context	37
4.3.3	Usability	37
4.3.4	Lifelong Learning	37
4.3.5	Compassion	38
5	Usability Study	39
5.1	Background	39
5.2	Type of Usability Study	39
5.3	Subject Population	40
5.3.1	Subject Rights	40
5.3.2	Subject Risks/Discomforts	40
5.3.3	Procedures to Maintain Confidentiality	41
5.4	Location and Time	42
5.5	Structure	42
5.5.1	Introduction	42
5.5.2	Usability Evaluation	43
5.5.3	Focus Group	43
5.5.4	Conclusion	43
5.6	Evaluation	43
5.6.1	Performance Metrics	44
5.6.2	Self-reported Measures	45
5.7	Results	45
5.7.1	Performance Metrics	45
5.7.2	Self-Reported Measures	49
5.8	Analysis	52
5.8.1	User Behavior during Usability Evaluation	52
5.8.2	Task Success, Time on Task and Efficiency	53
5.8.3	User Satisfaction	55
5.8.4	Research Goals	56
6	Conclusion	58
6.1	Future Work	58
6.1.1	Application Improvements	58
6.1.2	Integration into Educational Programs	60
6.2	Lessons Learned	60
6.3	Project Assessment	61
6.3.1	Disadvantages	61
6.3.2	Advantages	62
6.3.3	Objectives Met	62
A	Recruitment Flyer	63
B	Usability Evaluation Introduction	64
C	Usability Evaluation Survey	65
C.1	Basic information about yourself	65
C.2	Your experience with programming games	65
C.3	Code Girl usability evaluation	66

D Focus Group Outline	67
D.1 Research Questions	67
E Complete Usability Survey Results	68

List of Figures

2.1	State Diagram	6
2.2	General Application Activity Diagram	7
2.3	Building an Avatar Activity Diagram	8
2.4	Completing a Challenge Activity Diagram	9
2.5	Signup and Login Page Mockup	10
2.6	Main Page Mockup	11
2.7	Blockly Interface	11
2.8	Use Case Diagram	12
2.9	Architectural Diagram	15
2.10	Grace	21
2.11	Ada	22
3.1	Application Homepage	23
3.2	Avatar Customization Overview	24
3.3	Toolbar and Blocks	25
3.4	Color Block	26
3.5	First Challenge	27
3.6	Second Challenge	28
3.7	Third Challenge	29
3.8	Fourth Challenge	30
3.9	Angle Block	30
3.10	Fifth Challenge	31
3.11	Sixth Challenge	31
3.12	Seventh Challenge	32
3.13	Save the Avatar	33
5.1	Usability Evaluation Survey Results	50
5.2	Perception of Challenge Difficult by Age	56

List of Tables

4.1	Risk Table	35
5.1	Study Timing	42
5.2	Focus Group Structure	46
5.3	Usability Evaluation Results	47
5.4	Average Time on Task	47
5.5	Efficiency	48
5.6	Sub-task Efficiency	48

Acknowledgments

First, I would like to thank my partners on this project, Tracey Acosta and Paige Rogalski. From the time we came up with the idea for this application for our Senior Design Project, we all invested so much time, energy, and passion into making it as successful as possible. The application would not be what it is today without their help designing, implementing, testing, and refining the application during the first year of development. Much thanks to Paige for her continued help coordinating and leading user testing this past year. Conducting two users studies was a substantial undertaking, and would not have been possible without your assistance recruiting participants, mediating both the usability evaluation and focus group, and helping ensure reliable data could be collected and then later analyzed. We also would not have looked nearly as impressive during our Senior Design Presentation without your suggestion of matching *Code Girl* t-shirts, which were undoubtedly a hit.

I would also like to thank the fourth member of our team, our advisor at Santa Clara University, Darren Atkinson, for all of his support and encouragement over the past two years. You pushed us to design and develop *Code Girl* according to high software engineering standards from the beginning, and your feedback and guidance over the past year has been invaluable in helping me transform *Code Girl* into a thesis caliber application. You always kept us on-track and challenged us to research, seek out, and develop the best technology to help us achieve our goals. From issues such as determining whether to use a framework or not, to proof-reading every draft of our Senior Thesis and later this Master's thesis, to critiquing our presenting, to choosing a name for our application, you gave us advice that helped us better ourselves and our work. Without your guidance and feedback, we would not have accomplished all that we did, like winning Best in Session during our Senior Design Presentation, and *Code Girl* would not be the application that it is today.

Many thanks to Dr. Silvia Figueira at Santa Clara University for initially advising us on a project idea and her continued support throughout the process as well. Your suggestion to create an application that allows users to customize an avatar like a paper doll piqued our interest and sparked the idea of using customization to appeal to girls and engage them in learning. We built the story of *Code Girl* around this avatar and created an empowering figure that captured the interest of users and industry professionals alike. Also, your feedback on both our Senior Design Report and this thesis once was valuable in helping us initial developers and later just myself clearly and effectively communicate our work on *Code Girl*.

Lastly, I would like to thank my friends and family for their support as I worked on this project. You all entertained my many questions, from opinions on Grace's pose to accessories to challenge tasks, and helped me make the application better. Many thanks to the friends and family members who also came to our Senior Design Presentation on *Code Girl* to encourage and support us.

Chapter 1

Introduction

1.1 Problem Statement

Computing continues to grow in importance, yet few girls pursue this field. Today, fewer than one percent of women in college are majoring in computer science, making computing a heavily male-dominated field [1]. Girls have the potential to contribute immense creativity and ingenuity to the future of computing and technology, but few of them are introduced to opportunities to get involved. Exposing girls to programming at a young age fosters their interest in the subject, so that when they are older they may consider pursuing a career in computing [23].

In an attempt to address this problem, many educational programs designed to introduce girls to computing have been implemented in schools and camps across the country. Non-profit organizations such as *Girls Who Code* hold summer coding programs that educate high school girls about computer science and the skills they need to pursue computing [2]. In addition, programs like *Scratch* have been developed to begin teaching children aged eight and older how to code through games. *Scratch* and similar programs such as *Alice* have children use a visual coding editor to solve puzzles and create games, improving their ability to reason through problems and design projects [3] [4]. *Made With Code* recently launched a series of *Scratch*-like applications aimed at engaging girls in coding by allowing them to customize a picture of themselves, draw an avatar, design a bracelet and more with code [5]. These interactive and customizable coding environments pique the interests of children by giving them control and the ability to create real products.

Although educational programs, such as those organized by *Girls Who Code*, and visual-based programs like *Scratch* and *Alice* interest children and teenagers in computing, they are not enough. Educational programs are usually designed for children in middle school and up, and applications like those created by *Made With Code* are geared toward girls eight and older. Few programs exist for children younger than eight and those that do, such as *Tynker* games, are either aimed at boys or are designed to be unisex, which make them uninteresting to girls [6]. As such, girls have less exposure to coding from an early age and become interested

in other areas, perpetuating the lack of women in computer science and related fields.

Our solution, *Code Girl*, introduces girls ages five to eight to computing through an educational computer-science game designed to best appeal to their general interests while simultaneously teaching them coding. Many young girls love to play with dolls and spend hours dressing their dolls up, so we created a web-based application that allows girls to customize their own avatars using a visual coding editor. To unlock features and accessories for their avatars, the girls have to complete various challenges and puzzles designed to develop their abilities to solve problems using code. For example, before a girl is able to put shoes on her avatar, she has to successfully move an object through a maze by putting together blocks of code. We included a story-telling aspect in our application by making the avatar a superhero named Grace, whom the user then dresses up to assume any identity she chooses. Grace also has a robot sidekick named Ada who is incorporated into many of the challenges, giving the application a cohesive superhero theme and story that will engage users. To allow users to progress through the application at their own pace, girls can create an account, with their parents permission, so they can save their progress in designing their avatars at any point in the game. When they are finished creating their avatars, the girls are able to save their avatars as an image, which they can then print out to share with their friends, hopefully inspiring others to learn more about *Code Girl* and consequently computer-science. Given the young age of our users, we designed our application to be simple, colorful, easy to read, and easy to interact with, making it as user-friendly and engaging as possible.

We assessed the success of our project by partnering with Girl Scouts and analyzing how interested the girls were in learning more about programming after using our application. By designing an application specific to the interests of five to eight year old girls, we hope to interest this underrepresented group in coding from a young age and inspire them to pursue computing later on.

1.2 Background

As the initiative to introduce young children to programming grows, so do the number of available tools. Before designing *Code Girl*, we investigated a few of the previously mentioned platforms and applications, along with other popular online games for children, to understand and assess what they do and do not offer.

The first application, *ScratchJr*, is an iPad App where children ages five to seven can program their own interactive stories and games by dragging and dropping different puzzle-piece shaped blocks. Although *ScratchJr* has the customization options for the age group we are looking for, it does not specifically target young girls and limits the users to iPads. This will not work for our project because we want young girls to feel comfortable using a computer, not an iPad, through our application [7].

Code.orgs Studio is another platform that teaches basic computer skills through games and puzzles de-

signed for ages four to six plus. Course 1 and Course 2, the twenty-hour courses geared towards our target age group, are specifically designed to be taught in classrooms and include full lesson plans for the teacher. As a specific curriculum, *Code.orgs Studio* is designed to be unisex in an attempt to appeal to an entire classroom [8]. Our project hopes to captivate girls and motivate them to play and learn outside of the classroom.

The hardware and software combination of *Leapfrog* is well known for its educational games that are designed to begin teaching kids math, problem solving, language and more. An appealing attribute of *Leapfrog* is the use of popular cartoon characters in its games and its diverse selection of products and games for a range of ages. However, *Leapfrog* currently only has one game to teach computer programming and the characters are vikings, which appeal more to boys [9]. The age group for *Leapfrog* games aligns with our target audience, but the content is not geared towards teaching basic computer science principles.

MadeWithCode is an initiative by Google to get more girls in middle school and above excited about coding. On their website, they have a few beginner projects that use Blockly, a programming language made up of puzzle-piece shapes, to do tasks such as create an avatar using shapes, connect together virtual instruments to create a song, decorate a picture, or animate your own GIF [5]. Since *MadeWithCode* is one of the few platforms geared towards girls, we used some of their design ideas to appeal to a younger audience such as the colors and the simplicity of using Blockly to piece things together.

Another application, *Polly Pockets* online games, is a series of games featuring the main character and her friends in various locations and situations [10]. A popular doll, with her own elaborate and engaging online world, Polly Pocket inspired the idea of customizing a character and creating a story-based gaming environment. The downside of *Polly Pockets* games though, is that there is too much of a focus on activities such as shopping and less focus on educational enrichment, which is the main objective of our project.

These platforms all have something to offer in their own way, yet none solve the problem of specifically engaging girls ages five to eight in basic computer and problem solving skills. We thus created our requirements by taking the advantages of each of these platforms and combining them into a unique solution.

1.3 Requirements

For our solution, we identified and met the following functional and non-functional requirements, which describe the necessities of our application. The set of functional requirements define what must be done by the system, and the set of nonfunctional requirements define the manner in which the functional requirements need to be achieved. In addition, we identified and met design constraints, which are similar to non-functional requirements, but limit the way the solution is designed and implemented.

1.3.1 Functional Requirements

The functional requirements for our solution are summarized below. Our application is interactive, as users snap together blocks of code to customize an avatar and immediately see the results of their actions, in addition to receiving dynamic instructions on how to play the game. Users also connect blocks of code to solve challenges, which upon successful completion unlock new features for the users avatar. The avatar creation and challenges teach users basic logic and computer science skills, such as iteration, loops, and conditional statements. To allow our users to progress through the application at their own pace, girls can create an account, with their parents permission, so that they can save their progress in designing their avatar and completing challenges and return to the application later. When they are finished customizing their avatar, the girls can save their avatar as an image, which they can then share with their friends and inspire others.

1. The application is interactive.
2. The user snaps together puzzle pieces to customize an avatar.
3. The avatar created by the user is saveable.
4. The application provides instructions.
5. The user solves challenges by snapping together puzzle pieces.
6. The user unlocks new accessories for her avatar by successfully completing challenges.
7. The application saves user progress.
8. The application teaches basic logic skills including positioning, loops and sequences.

1.3.2 Non-functional Requirements

In addition to functional requirements, we outlined and achieved the following non-functional requirements, summarized below. Given the young age of our users, we designed our application to be colorful, easy to read and easy to interact with, making it as user-friendly, engaging, and intuitive as possible. Our application is also compatible with desktops, laptops, and tablets, making it competitive with applications currently available.

1. The application is user-friendly.
2. The application is intuitive.
3. The application is compatible with desktops, laptops and tablets.

1.3.3 Design Constraints

With regards to design constraints, summarized below, we identified and met only two. First, our application is web-based because of undergraduate degree requirements, but this has the advantage of making our application widely available to potential users. Most importantly though, given the young age of our users, our application is compliant with the Children’s Online Privacy Protection Act (COPPA), which constrained the manner in which we gathered and stored user information.

1. The application is web-based
2. The application is compliant with the Children’s Online Privacy Protection Act [11]

1.4 Conclusions

Through research into existing applications, we were able to take the design and learning outcomes we thought were most beneficial for engaging and educating young girls and craft them into requirements for *Code Girl*. The functional requirements include the main activities the user is able to do, as well as what we want to teach young girls through playing with our application. These requirements and ideas then gave way to designs of how these activities fit together and how the user might interact with the system.

Chapter 2

Design

2.1 Conceptual Model

Before implementing our application, we created conceptual models, in the form of state diagrams, activity diagrams, and mockups, to help us determine and visualize the flow of system and the design of the application.

2.1.1 State Diagram

From the functional requirements, we determined the main states of the system and how they interact with each other. The state diagram, Figure 2.1, shows the two states and how the system transitions from one state to another. At any given time, the user can only be in one of these states. For example, while a user is playing a challenge, she cannot simultaneously edit her avatar. The user must successfully complete a challenge in order to unlock a new challenge and accessories. Once a challenge is successfully completed, the user is taken back to the avatar, which she can continue to customize.

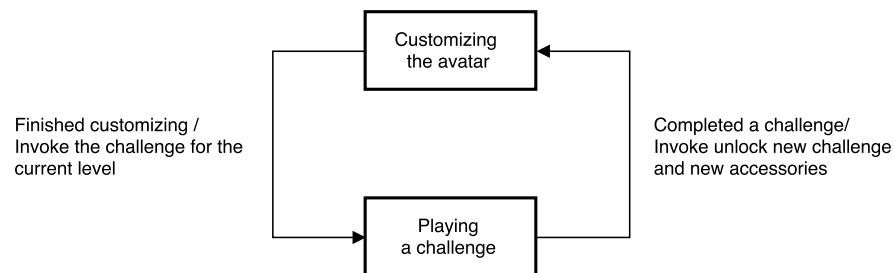


Figure 2.1: A state diagram that shows the two main states of the system.

2.1.2 Activity Diagrams

Each state is made up of many actions. An activity diagram combines these actions into main activities done by the user. The first activity diagram, Figure 2.2, focuses on showing the overall activities of a user of our system. Once a user accesses the system by signing up or signing in, she is taken to the main avatar customization page, where she can build her avatar. On this page, there is a progress bar indicating if the user has challenges available to play. If she does, she may successfully complete them to unlock accessories for her avatar. At any given point, the user can save her progress and continue playing or log out.

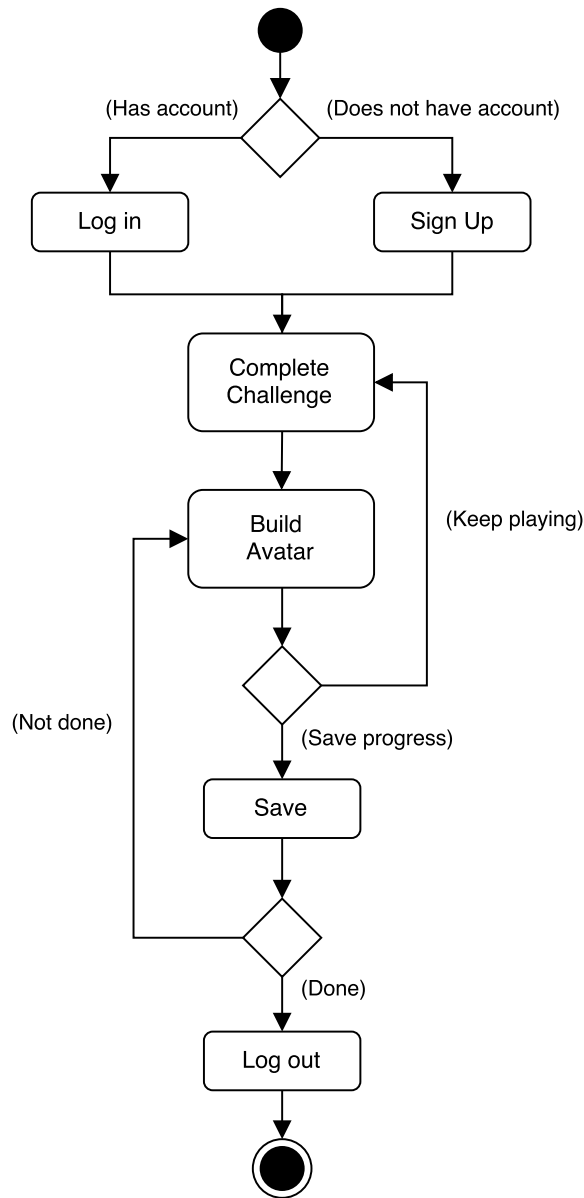


Figure 2.2: The activity diagram of the general application.

While the general activity diagram of the application gives a high-level view of the activities, it does not go into detail about how the user builds an avatar or completes challenges.

Figure 2.3 shows the actions a user takes to create an avatar. These actions focus on the user interaction with puzzle-piece shaped blocks. As the user selects, drags and drops, and arranges these blocks on the canvas she can see the customization of her avatar taking place. If the user wants to try and unlock new accessories, she can choose to play a challenge; otherwise, she can continue to edit her avatar.

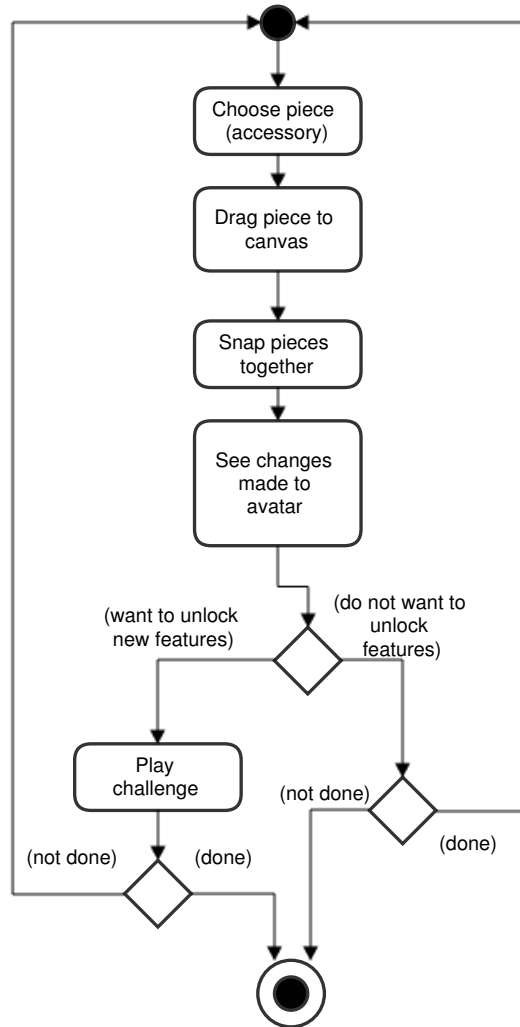


Figure 2.3: The activity diagram of building an avatar.

Figure 2.4 depicts the flow of activities of a user completing a challenge. In this example challenge, the user arranges puzzle-piece blocks to direct their character through a maze. The blocks contain a single instruction, which may tell the character to turn either left or right or to move either forward or backward. Each instruction is translated into code that is then run to move the character through the maze. If the user successfully completes the challenge, she unlocks new accessories for her avatar. Otherwise, she can reset the character to the beginning of the maze and arrange the blocks in a new combination, adding and removing instructions as necessary.

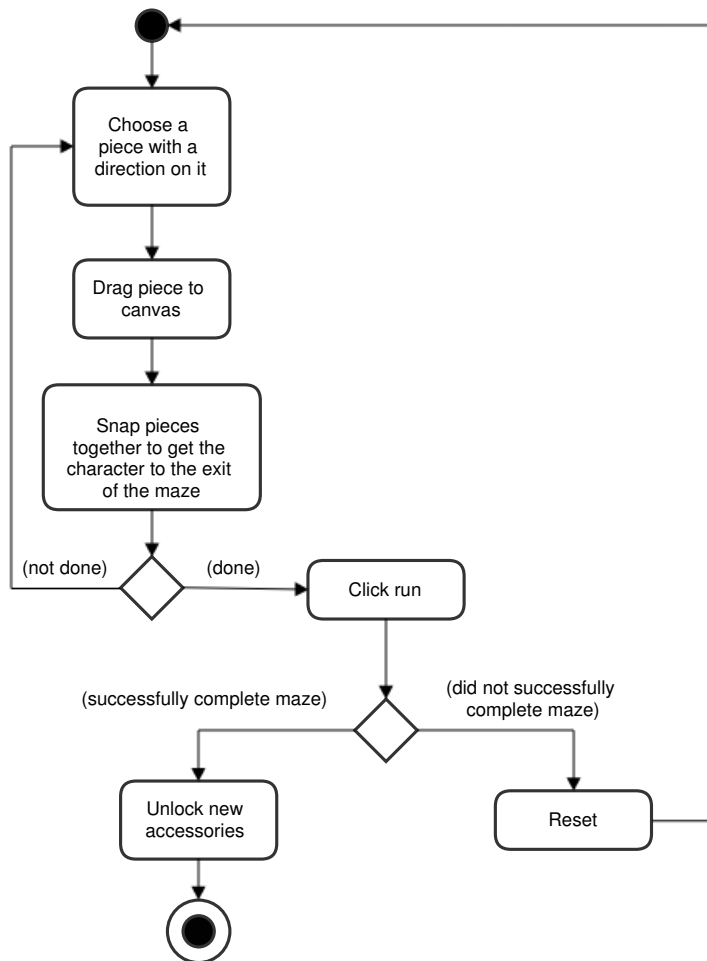


Figure 2.4: The activity diagram of completing an example challenge.

2.1.3 Mockups

Low fidelity mockups provided us with a way to show the user-interface design to visualize how the user will accomplish the main activities in our application and gave us a framework to begin designing and implementing the game. Figure 2.5 shows an initial idea for the design of the login or signup page. We decided to

combine these actions into a single page since they relate to the user’s account. It is also important to note that because our target audience consists of minors, we ask for the parent’s email to make sure the parent gives permission to join, thereby making our application compliant with the Children’s Online Privacy Protection act, as specified in our design constraints.

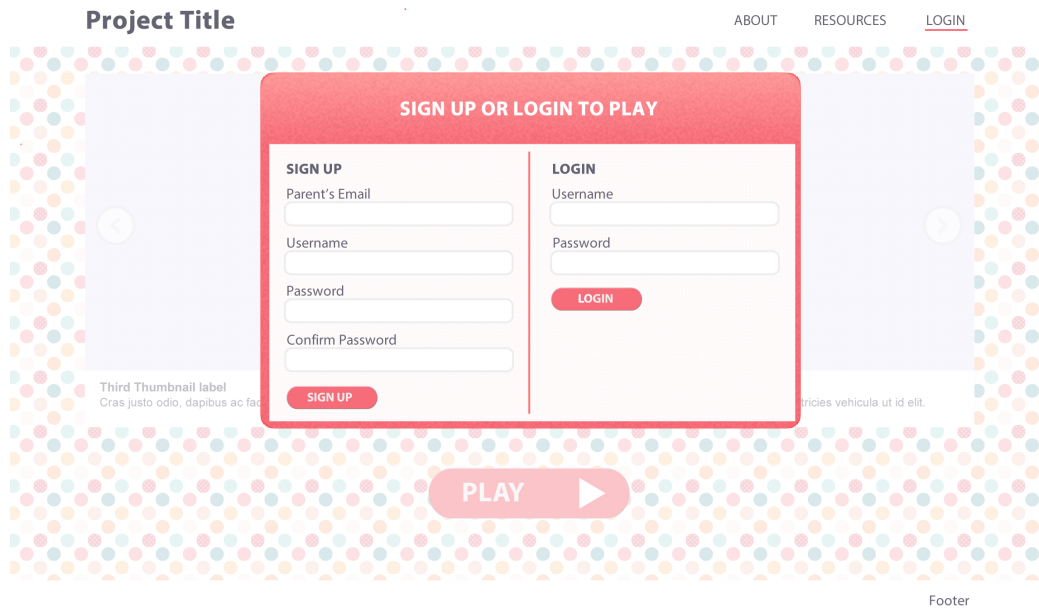


Figure 2.5: A mockup of the signup or login page.

The mockup of the main page of *Code Girl*, where the user will create the avatar, is shown in Figure 2.6. The main page is divided into three sections. To the left is a library of available blocks that the user can use to create the avatar. To the right of the library is the canvas where the user snaps the pieces together. On the far right is where the avatar is shown, so the user can see the changes made as she customizes her avatar.

The mockup of the avatar customization page was derived from the standard Blockly environment layout, as shown on the library’s developer pages and depicted in Figure 2.7. Using terminology from Blockly documentation, which is used throughout this report, the three main components of the interface, as referenced above, are: (1) the Toolbar, (2) the Blockly Canvas, (3) and the Visualization box. The Toolbar consists of the categories users select blocks from. The Blockly Canvas is where users drag blocks to, so they can move them around and/or connect them to other blocks currently on the canvas. The Visualization box displays the results executing the code represented by the blocks on the Blockly Canvas. Each block represents a chunk of Javascript code that when moved on the Blockly Canvas, is executed and displayed in the Visualization box.

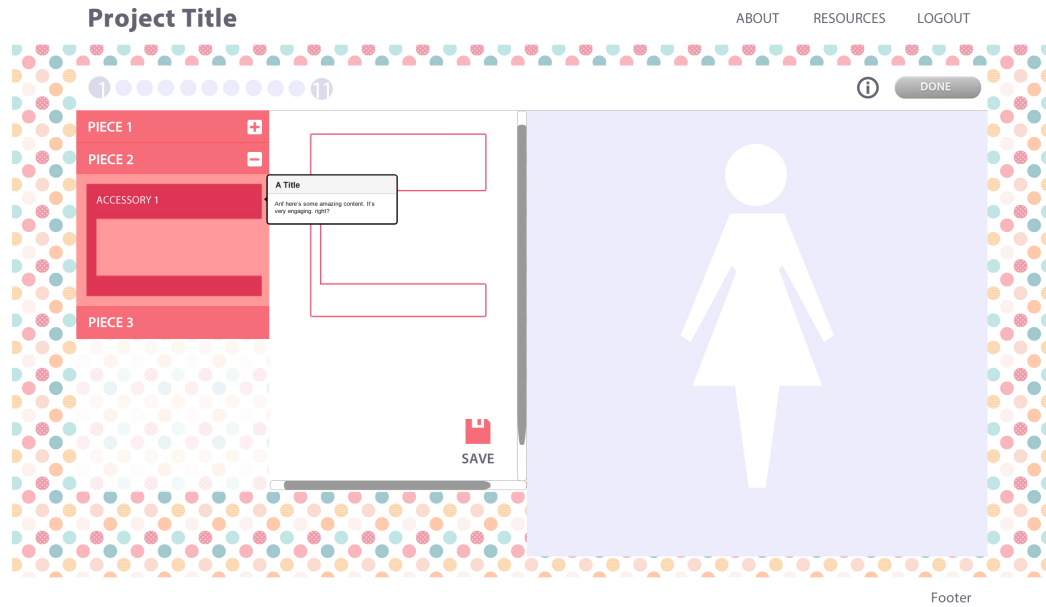


Figure 2.6: A mockup of the main page.

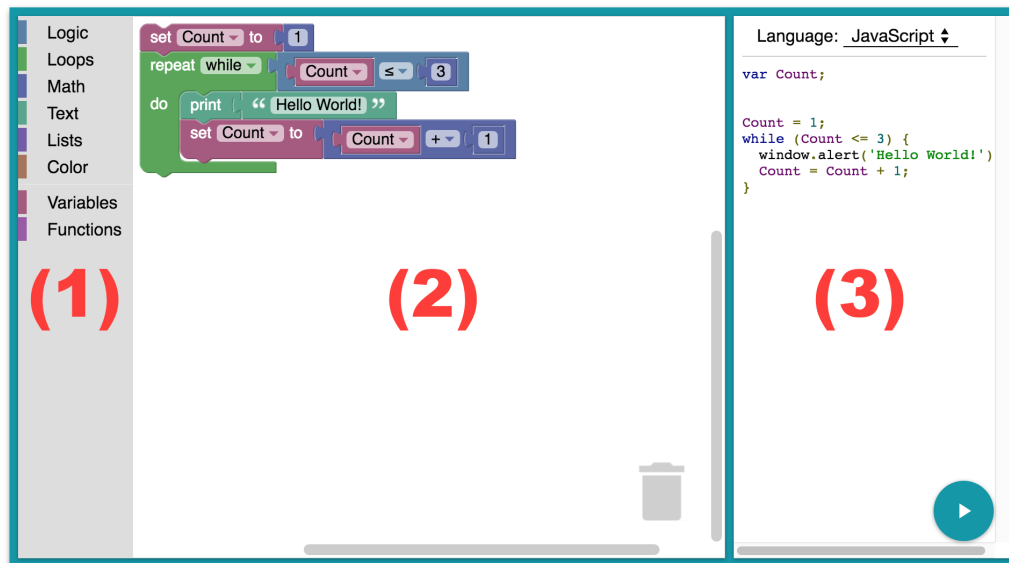


Figure 2.7: The three main components of the standard Blockly interface [12]

2.2 Use Cases

A use case defines the steps required to accomplish a specific goal, as described in the activity diagrams and visualized in the mockups. The following use cases describe how the user interacts with the system to achieve these goals, including conditions that need to be met, steps required, and common errors that might occur. The use case diagram, Figure 2.8, helps to illustrate the major actions the user will take when using

our system.

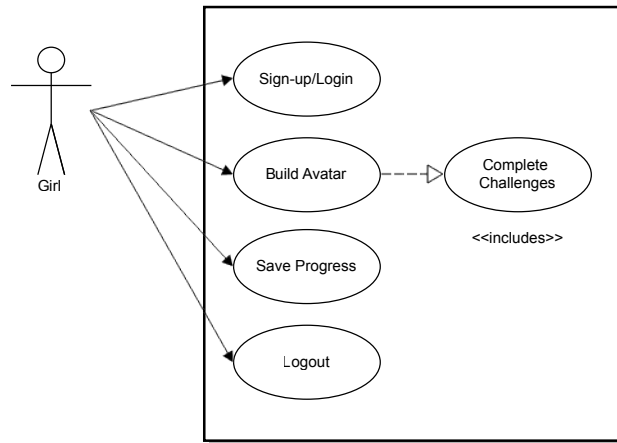


Figure 2.8: The use case diagram.

1. Signup/Login

- Goal: Sign-up or login to begin building the avatar
- Actor: Girl (with Parent)
- Precondition: The user does not have an account if she wants to sign-up and the user has an account if she wants to login in.
- Postcondition: The user is logged into the system.
- Scenario:
 - (a) The user either selects the “SIGN UP” or “LOGIN” button.
 - (b) The user either fills out the form to sign up or the form to login.
- Exceptions:
 - (a) The user wants to sign up, but her email is already in the system.
 - (b) The user forgets her password to login.

2. Build Avatar

- Goal: A customized avatar/doll has been created
- Actor: Girl
- Precondition: The girl has unlocked all of the features and built her customized avatar.
- Postcondition: The girl has built her own avatar using blocks of code.

- Scenario:
 - (a) The user chooses a piece that she wants to add to her avatar.
 - (b) The user drags the piece to the canvas.
 - (c) The user snaps the piece together with the other pieces on the canvas.
 - (d) The user sees the changes made to the avatar.
 - (e) The user plays challenges to unlock more pieces.
- Exceptions:
 - (a) The user wants to skip a challenge.

3. Save Progress

- Goal: The user saves the blocks she has put together to build her avatar.
- Actor: Girl
- Precondition: The girl has at least one block on the canvas for building her avatar.
- Postcondition: The girl's progress has been saved.
- Scenario:
 - (a) The girl clicks the "SAVE" icon.
- Exceptions:
 - (a) None

4. Logout

- Goal: The user is logged out of the application.
- Actor: Girl
- Precondition: The girl is logged in.
- Postcondition: The girl is logged out.
- Scenario:
 - (a) The girl clicks on the "LOGOUT" button.
 - (b) The girl is prompted to save her work.
 - (c) The girl confirms she wants to logout.
- Exceptions:
 - (a) None

2.3 Architectural Design

The activity diagrams, mockups, and use cases all present how the user interacts with the system. An architectural diagram shows the components and connections of the system as a whole. Figure 2.9 displays a multi-tier architectural style that models our system [22]. Although Blockly is primarily client-side, a server and database are still necessary for user-session logic and storage.

The system is divided into four main modules, described in detail below, to allow different components to be managed independently, a modular programming technique. In modular programming, a program is divided into distinct, independent modules, each of which completely covers one piece of the application's functionality [22]. Isolating the modules helped us with development and testing that the challenges and avatar worked separately before we integrated them.

2.3.1 Avatar Customization

The first module focuses on the avatar customization component of the game. This includes the blocks used to create the avatar and customization logic. The blocks are the custom clothes and accessories blocks, as well as the blocks that allow users to change Grace's physical features, such as her hair color. The customization logic refers to the functions that translate the blocks on the user's avatar to code that is then used to render physical characteristics, outfits, and accessories on the avatar. This module also includes the SVG graphics for all of the physical characteristics, like hair style, the clothes, and the accessories designed for the game.

2.3.2 Challenges

The second module contains the challenges the user can play and the blocks relevant to these challenges. The game initially included only four challenges, but was expanded to include seven challenges, which often build on the concepts taught in previous challenges. Each challenge uses its own set of custom blocks, while also relying on blocks, such as color blocks, used by other challenges in this module, an unavoidable instance of coupling between challenges.

2.3.3 Game Logic

The game logic module defines the integration between unlocking challenges and new clothes or accessory blocks. Progression through the game is pre-determined and fixed, meaning that upon completing Avatar Customization Level X the user is always then taken to Challenge X, never Challenge Y or Z, and so on. The interaction between avatar customization levels and challenges is managed on the client side to provide flexibility and fast responses, so successful progression through the game can be tracked and managed without needing to constantly make server requests, which could slow the application down.

2.3.4 Loading Logic

A server, however, is used in the fourth module, which defines the logic for saving the user's progress in the game. Since users should be able to play the game at their own pace, they may want to temporarily stop playing the game and resume playing later on, necessitating both the concept of a user session and logic for restoring a user's session on subsequent logins. As such, the loading logic module contains the functionality for determining if a previous session exists for a user and either returning the user to the avatar customization level or challenge she was most recently one, with the blocks she was using at the time of last logout, or bringing the user to the first challenge of the game.

2.3.5 Database

By hosting our application on Google App Engine we have access to Google Cloud Storage to save and load the program, as well as keep track of user accounts and authenticate our users. Account information is stored in a database, along with the blocks corresponding to the user's progress in the game. User account and user session queries are made to the database via the survey.

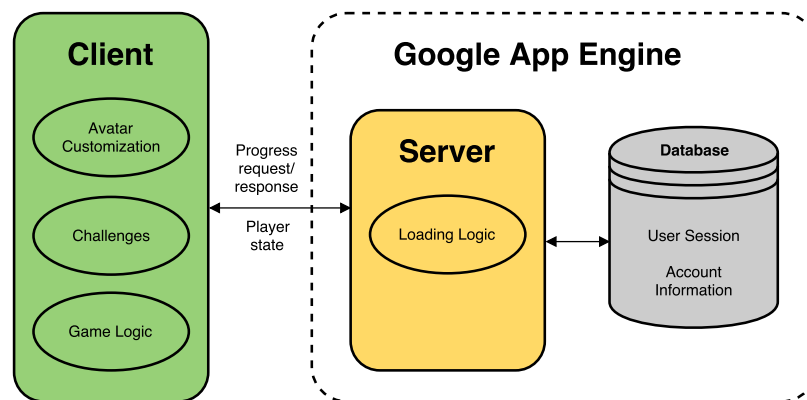


Figure 2.9: An architectural diagram of the system.

2.4 Technologies Used

To accomplish our design and achieve the desired functionality previously described, we used the following web-development technologies and services:

- Blockly: A client-side library that can be used to build programs by snapping together puzzle-piece blocks in a visual editor created by Google. Blockly provides us with the framework for a drag and drop environment and the library of coding puzzle piece blocks used in the games [12]. The Blockly environment was used to develop the application Blockly Games, that's open source games were the basis for some of the challenges incorporated in *Code Girl* [13].

- **HTML5:** The most recent version of the standard HTML markup language for the web, as introduced in 2008. HTML5 introduced semantic elements like `<header>`, which are used in code for this application to make it easier to maintain and understand. HTML5 also introduced graphic elements, such as `<canvas>` and multimedia elements such as `<video>`. In this application, the `<canvas>` element is used to render SVG graphics on the page. These graphics are drawn dynamically via JavaScript in response to changes made by the user. The `<video>` element is used in this application to provide video instructions for different levels [15].
- **JavaScript:** A programming language that enables client-side interaction with the user, browser control, and changes to the content displayed on the webpage. Blockly blocks can be easily exported to JavaScript, supporting a dynamic, online gaming environment [12]. In addition to Blockly, the following JavaScript plugins were used:
 - **html2canvas:** A fully client-side script that enables developers to capture a image of a webpage, or a section of the page. This image is built by traversing through the DOM of the page after it is rendered and creating a canvas from the elements, including the CSS styles. A specific portion of the page can be selecting by specifying which DOM element to traverse [28]. For this application, the Visualization box containing the user’s customized avatar, a stylized `<div>` element with a `<canvas>` element inside, is captured and resulting canvas image is converted into PNG image using the `HTMLCanvasElement.toDataURL()` method, which returns a URL to the image [29].
 - **download.js:** A client-side script for downloading images using JavaScript and HTML5, supported by many devices and browsers [30]. The script is used to directly download the PNG image URL, generated as described above, as a file called `CodeGirl.png` to the user’s device.
 - **BxSlider:** A fully responsive jQuery content slider used to display images on the homepage. BxSlider is compatible by Firefox, Chrome, Safari, iOS, Android, and Internet Explorer, and supports both touch and swipe movements, making it suitable for running the application on a variety of devices [31].
- **jQuery:** A JavaScript library that implements common JavaScript functionality to simplify client-side scripting [14].
- **XML:** A markup language that specifies how a document is encoded. XML is used in this application to save the configuration of blocks on avatar customization levels and challenge levels between user sessions.

- Google App Engine: A Platform as a Service that enables developers to build and deploy an application using Google's runtime and development environment. Google App Engine provides many features, such as data storage, retrieval, and search [16].
- NoSQL: A database that enables the storage and retrieval of data, such as account information, but does not use tabular relations [18].
- Python: A scripting language that we used to communicate between the Google App Engine and the database [17].
- GitHub: A web-based repository service used for revision control and source code management [19].
- Adobe Illustrator: A software program used for designing and editing graphics, which can then be saved in a number of formats, such as PNG and SVG, as were used in this application.

2.5 Design Rationale

Before making any decisions regarding the design and implementation of our application, we thoroughly researched how to effectively design games to teach computer science and tested successful applications, such as *Scratch* and *Made with Code*. Our design was thus driven by our goal of educating young girls in a fun and interactive way to inspire them to pursue computer science or technology. To achieve this goal and the requirements we identified, we made several design choices, presented and discussed in this section.

2.5.1 Game Design

We decided to create a game-based application to inspire young girls to pursue computing because game-based learning through applications such as *Scratch* and *Alice* has been proven effective in teaching children computer science concepts and skills [21]. In our application, users snap together pieces of code to create their own avatar and complete challenges that unlock more features for their avatar within the context of a larger story. We chose these activities because research shows that educational games are more effective when they actively engage the users with stories and interactive puzzles. Additionally, in researching non-educational games, such as *Polly Pockets*, that are popular among our target audience of five to eight year old girls, we noticed that many games allowed users to customize their character, indicating that personalization engages young girls. Incorporating challenges that unlock new features will not only engage our users by allowing them to customize their avatar, but will also serve as an incentive for our users to continue playing our game, and thereby learn computer science concepts and skills. By introducing young girls to computer

science in a fun and interactive game that is similar to the games they already play, we hope to teach them basic skills and concepts and inspire them to learn more about computing.

2.5.2 Visual Programming Environment

Traditional programming languages and environments are the most effective means of teaching computer science concepts and skills, but they are too complicated for young children to understand. They also are usually taught in a classroom setting or online environment, and difficult for people to learn on their own. Since our target audience is very young and presumably using our application on their own, we chose to use a visual programming environment to best meet their needs and skill levels. Visual programming environments are easier to learn in than traditional, language-driven environments, because they not require the users to compile their code or verify syntax. They essentially allow users to write error-free programs without any instruction by abstracting away the syntax of the code, which will benefit our users since their reading, writing, and reasoning skills are still developing. By removing the issues of syntax and coding errors, visual programming environments focus the users attention on the logic of the problem, introducing them to basic computer science concepts, such as loops. Additionally, using visual blocks that users drag-and-drop and snap together to build a program or solve a challenge facilitates interaction and engages the user creatively. The specific visual programming editor we chose to use is Blockly for the following reasons.

2.5.3 Blockly

Created by Google, Blockly is a JavaScript library that developers can use to build a visual coding editor that enables users to write programs by snapping puzzle-piece like blocks of code together. As the blocks are put together, the library generates and executes the corresponding code, showing the results to the user. We decided to use Blockly because of its benefits for both developers and users. Blockly, like *Scratch*, is open source, but Blockly is designed for developers to integrate into their own applications, whereas *Scratch* must be exported or embedded on external websites. Since Blockly is open-source and can be used in custom applications, we can extend its functionality to create an application personalized to our target audiences interests and skills. Blockly was also influenced by *Scratch* and *App Inventor*, so it is easy for young children to use, making it the best tool for engaging our target age group of five to eight year old girls [3]. Blockly, furthermore, meets the design constraint that our application must be web-based and gives our application portability, since it is compatible with Chrome, Firefox, Safari, Opera and Internet Explorer [12].

2.5.4 Technologies

Our application is built primarily using Javascript, making it a client-heavy application. Blockly, the visual coding editor we chose to use, is 100% client side and written in Javascript, which can then be compiled to Javascript, Dart, or Python, but we chose to only use Javascript, as we are all familiar with the language. To simplify the creation of our application and achieve consistency, we initially decided to use the AngularJS framework. We chose the AngularJS framework over similar frameworks such as Backbone.js and Ember because it is well suited to single page applications. Additionally, AngularJS has been used to develop Blockly applications and does not necessitate a dominant rest API application, which benefits our client-heavy application. AngularJS also does not have as steep of a learning curve as Ember does. Once we began implementation, however, we decided not to use AngularJS because Blockly provided us with a sufficient framework for building our application. When necessary, we used JQuery to simplify the Javascript that we needed to write, since it implements many common functions. To save the users blocks and then restore them when they return, we used calls to export to and return from XML. To enable users to save, load, and share their work, we hosted our application on Google App Engine, which provided these features that we could access using Python and the cloud storage API. Additionally, we used a NoSQL database, provided with Google App Engine, to save the user's progress. We, as such, enable our users to play at their own pace to maximize their learning and enjoyment.

2.5.5 Responsive Design

One of the non-functional requirements for *Code Girl* is that “The application is compatible with desktops, laptops and tablets.” We identified this as a requirement and designed our application accordingly in order to keep up to date with the growing popularity of mobile games and make our application competitive with programs like those developed by *Made with Code*. We, however, developed the application primarily on desktop and laptop browsers and screens, making the Blockly canvas fluid height and width so that it would resize on smaller displays, but did not fully test the responsive functionality. After early testing of the application, we realized the display was distorted when viewed on smaller screens. When the application was viewed on a small display, such as a tablet, the Blockly canvas resized to be so small that it was difficult for users to drag blocks onto it and connect them with other pieces. The non-functional requirement was thus not adequately met and improvements were made to make the application fully-responsive.

Additionally, as we realized in user testing, children are more comfortable using tablets to play games, further necessitating an appealing and intuitive responsive design. By making our application more compatible with tablets, we are engaging users in a mode they are more comfortable with, since as we learned in

user testing, children are less familiar using a desktop and mouse to play games than they are using mobile devices.

2.5.6 Story-telling

Research has shown that games that engage children and appeal to their interests are more effective at teaching computer science concepts and skills [21, 24]. Stories capture the interest of young children and engage them in learning and playing. As such, a superhero theme and story was incorporated into *Code Girl* via Grace and her sidekick Ada, as superheros are not only popular amongst children today, but are also inspiring and empowering figures.

Grace

The main character in the *Code Girl* story is superhero Grace, named after Grace Hopper, a female computer scientist who lead the team that developed the first compiler. Grace, pictured in figure 2.10 was custom designed to appeal to young girls, with the use of bright colors in her outfit and bold accessories, while also inspiring them to be creative and strong. The lack of diversity with Grace was identified when first designed, but not addressed until later versions of the application, when functionality was incorporated to allow users to change Grace's physical appearance. This functionality was built into the application to further empower girls, allowing them to see that anyone, regardless of their gender or ethnicity, could be a superhero, and hopefully inspire them to see themselves as the future of programming and technology.

Ada

Grace was given a robot sidekick, shown in Figure 2.11, named Ada after Ada Lovelace, who is often considered the first computer programmer. Ada, with both her name and by being a robot, maintains the technology-based focus of the game, while also furthering the superhero story, as many superheroes have their own sidekick. Ada is incorporated into many of the challenges to provide cohesion and further interest girls in completing the tasks of the various challenges, engaging them in the process of learning the computer science concepts that these tasks teach.

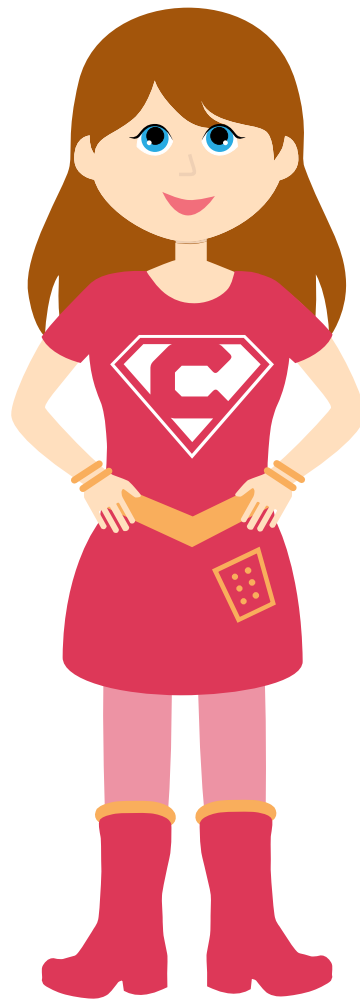


Figure 2.10: Grace, the superhero main character of the *Code Girl* story

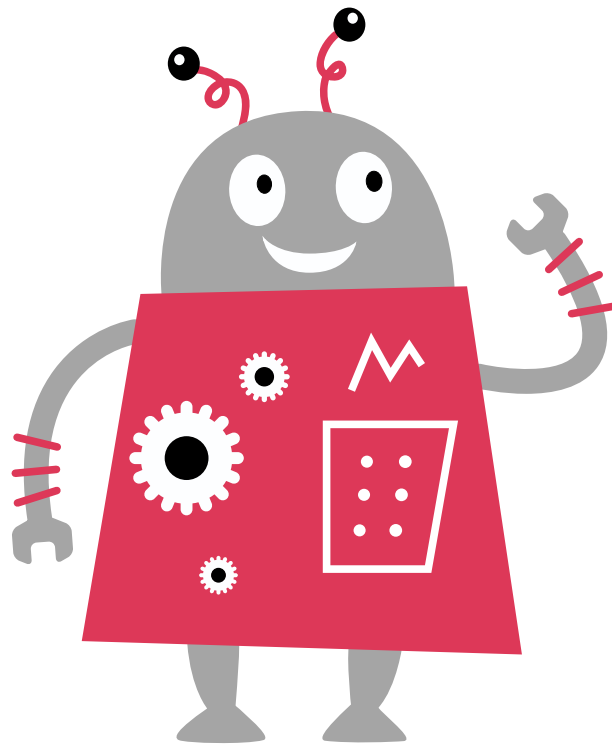


Figure 2.11: Ada, Grace's robot sidekick

Chapter 3

Application Overview

When users type in the URL for Code Girl, they are taken to the application’s homepage, pictured in Figure 3.1, which introduces them to the superhero story of the game. A slideshow presents Grace, the superhero a user can customize to assume an identity of her choice, Ada, Grace’s robot side-kick, and the objective of the game, which is to customize a unique avatar while learning computer science concepts. The homepage is designed to be simple, with minimal interactivity, while also capturing the attention of potential users and making them excited to play the game. Users must have an account to use the application, so their progress can be saved, enabling them to play at their own pace and return to the game later on. The game consists of seven avatar customization levels and challenges designed to teach logic and computer science skills, explained in detail the following sections.

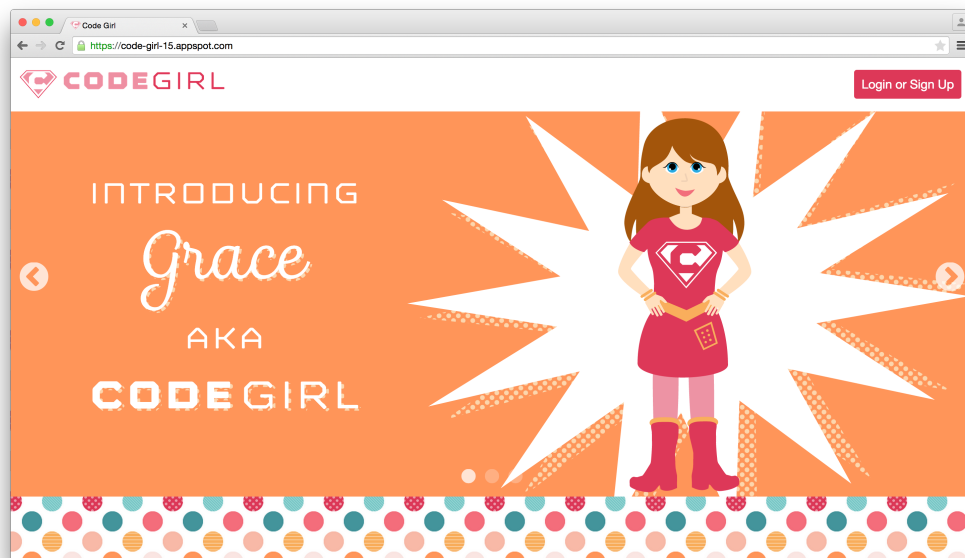


Figure 3.1: Users are initially taken to the application’s homepage.

3.1 Create and Account/Login

To begin using the application, users select the “Login or Signup” button on the homepage. Because our application is hosted on Google App Engine, user registration and authentication is performed through Google Accounts. Consequently, users are taken to the GMail login page, where their parents can sign up for an account or log in for them. Google Accounts requires account holders to be thirteen or older, thus by performing user sign on and authentication through GMail, we ensure that users have their parent’s or guardian’s consent and their privacy is being protected, a requirement of our application given the young age of users [27].

3.2 Avatar Customization

After users log in to the application and play the introductory challenge, discussed below, they are taken to the page for the first avatar customization level, show in Figure 3.2, where they can see their avatar and begin customizing her. The users can track their progress in the game via the level icons below the site title. Seven circles refer to the seven levels of the game. The user’s current level is displayed in a large gray circle. Previously completed circles are represented by small gray circles and levels yet to be completed are represented by small empty circles, except for the last level. The rest of the page is split into the Blockly toolbar and canvas on the left and the avatar visualization on the right. The toolbar consists of categories of avatar features and accessories, which when selected, display the blocks for that category, as shown in Figure 3.3.

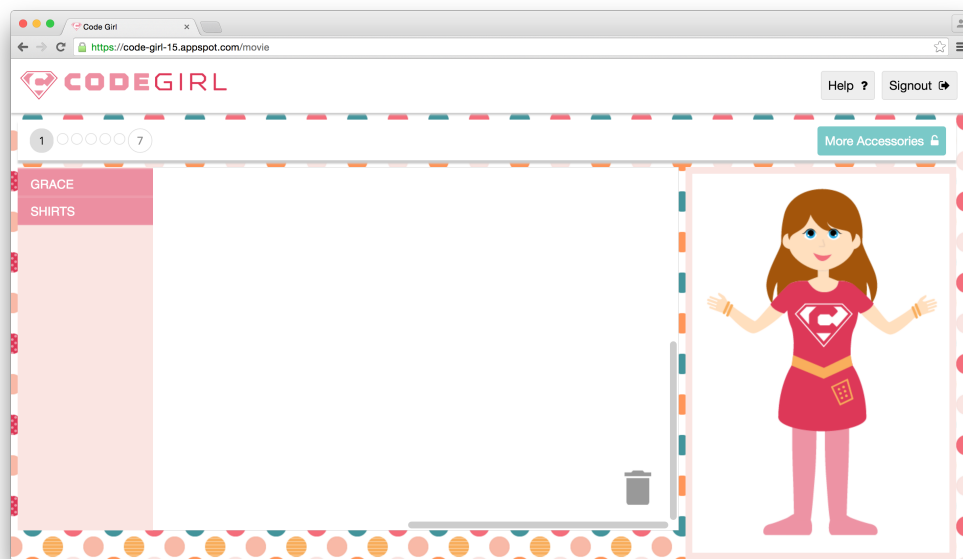


Figure 3.2: Users customize their avatar, on right, using the accessories from the toolbar.

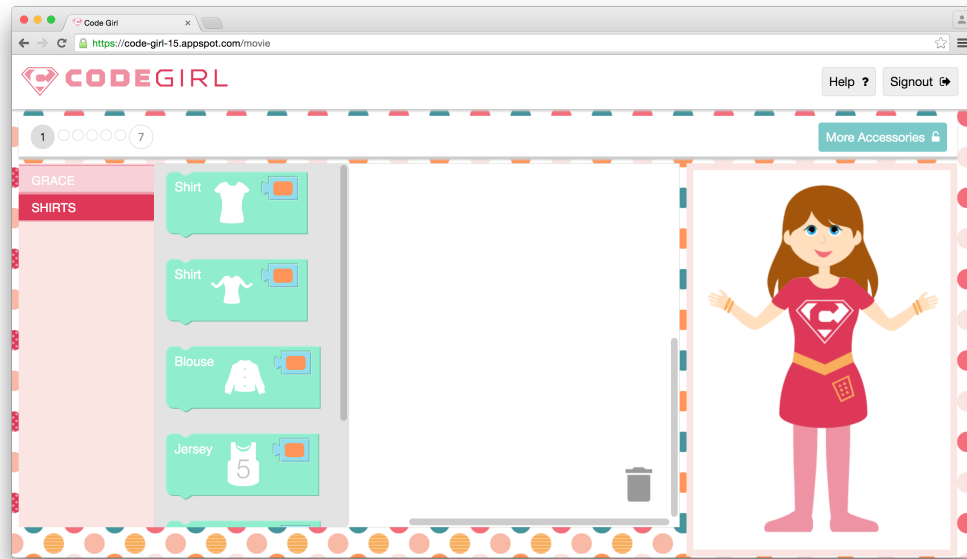


Figure 3.3: The blocks available for the shirts category in the toolbar.

Two categories, Grace and Shirts, are initially available to users. By selecting different blocks from the Grace category, users can change the avatar's features, which include hair style and color, as well as eye and skin color. The Shirts category is the first clothes/accessory category users have access to, and as the new clothes and accessories are unlocked, the categories for these items appear below in the toolbar.

To begin customizing Grace, users select a block from the expanded category menu in the toolbar and drag the block to the canvas. Users can then change the color of the feature, item of clothing, or accessory they selected by clicking on the color square embedded in the shirt block and selecting a new color from the popup menu, as shown in Figure 3.4. If users want to change or remove a feature, item of clothing or accessory they have added, they can drag the block to the trash in the lower right-hand corner of the canvas and select a different one from the toolbar.

A total of eight categories are available to users for them to customize Grace. The categories, designed to appeal to the interests of girls ages five to eight and empower them to be who they want to be are:

1. Grace
2. Shirts
3. Bottoms
4. Shoes
5. Accessories

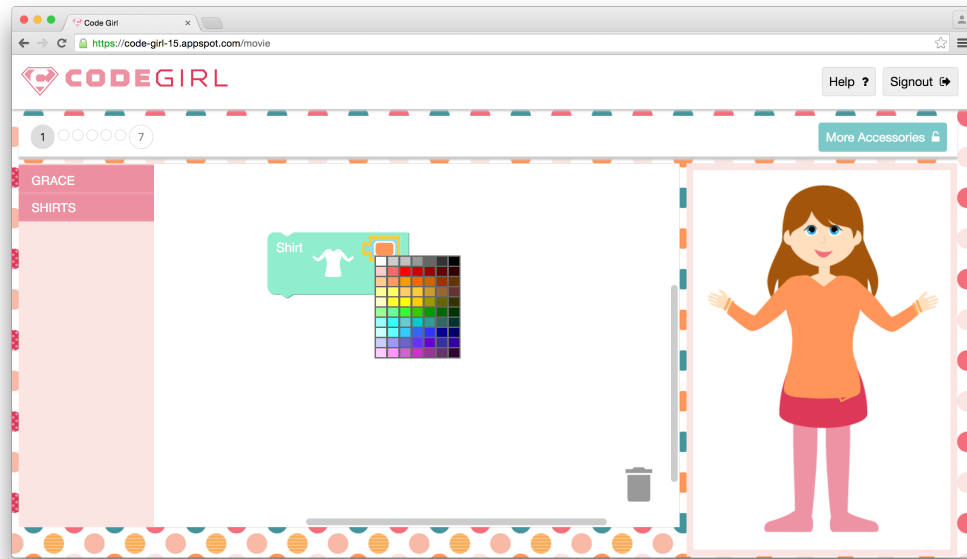


Figure 3.4: Changing the color of an item.

6. Bags
7. Princess
8. Superhero

The avatar customization process serves as a reward system to engage girls in the game and make them want to complete challenges, which introduce them to computer science and educate them as they play. In order to get more clothes and accessories for the avatar, so they can customize Grace further, users need to complete the challenges. To play the challenge for the level the user is currently at, the user selects the “Unlock Accessories” button above the avatar.

3.3 Challenges

The game contains seven challenges designed to introduce girls to Blockly and basic logic and programming concepts, described in detail below.

3.3.1 Challenge 1: Introduction to Blockly

Users are taken to the first challenge immediately after logging in or signing up, before the first avatar customization level, in order to introduce new users to the Blockly programming environment. For the first challenge, pictured in Figure 3.5, users need to complete a puzzle, where they connect a block that has the picture of an animal with the block that lists the animal’s name and select how many legs the animal has.

The purpose of this challenge is to familiarize users with Blockly blocks, introducing them to the concepts of making selections within the blocks and connecting blocks together to group related items. Once users think they have the correct answer, they can click the “Check Answer” button. If they are right, more clothes and accessories will be unlocked, and the users are redirected to the avatar page. If users did not complete the puzzle correctly, the number of errors will be shown, with the blocks that contain errors highlighted, and users will be prompted to correct their mistakes. Feedback is provided both via the error message and block highlighting to make errors both more noticeable and easier for users to correct.

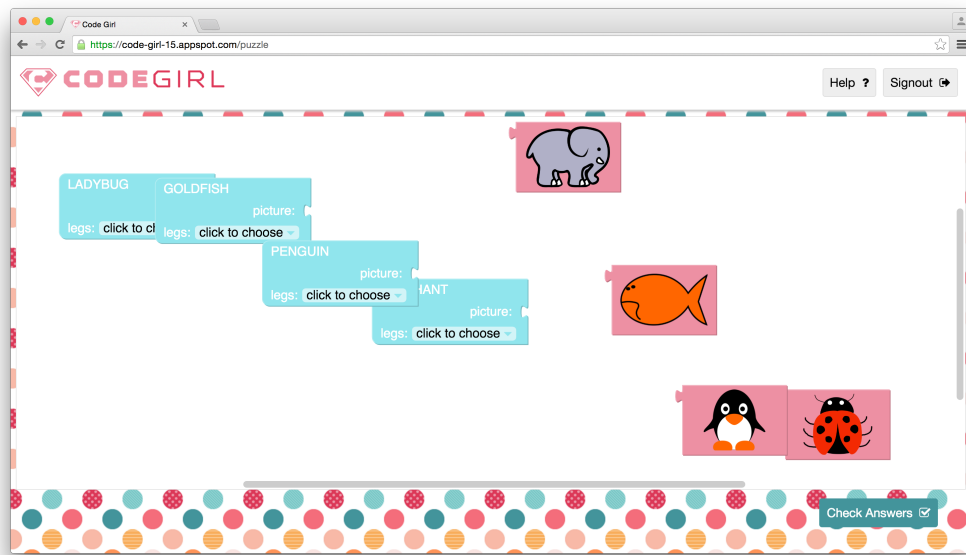


Figure 3.5: Users complete an animal puzzle for the first challenge to introduce them to Blockly.

Now, on the next avatar customization level, users can see a new category of blocks they can use to customize their avatar further. Upon completing the puzzle challenge, for example, and being taken to the second level in the avatar creation process, a new category, Bottoms, appears in the toolbar and users can then add bottoms to the avatar and change their color as well.

3.3.2 Challenge 2: Sequentiality

In the second challenge, shown in Figure 3.6, users begin to be introduced to logic and coding concepts, the first being sequentiality. We give users all of the blocks they need to complete the challenge on the canvas to begin with because we want users to focus on the importance of having logical steps, or in a more general sense code, in order, and not on deciding how many steps to take. For this challenge, users are given a brief story-based challenge, to help Ada draw a square as visualized in image and gif presented in the instructions. The instructions include both text and visualizations to accommodate all reading levels, so users

can understand the challenge even if they are not familiar with all of the words in the text instructions, a risk for younger users. Once a user thinks she has the correct answer, she can click the “Run Program” button and see the result of her code, with each block highlighted as the corresponding statement is executed.

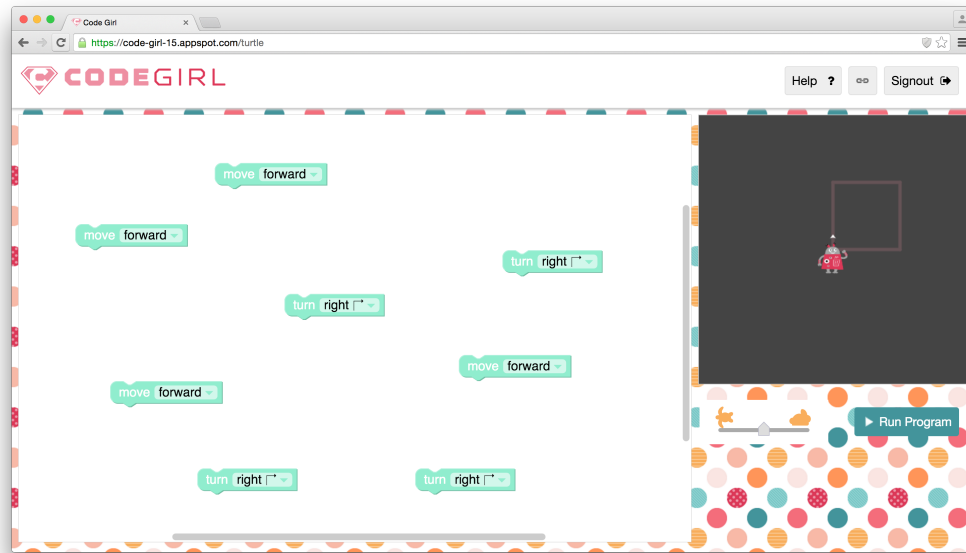


Figure 3.6: Users learn sequentiality by connecting the blocks of code on the canvas.

3.3.3 Challenge 3: Loops

Progressively increasing the difficulty of the challenges users are given to build on the computer science concepts taught, the following challenge, seen in Figure 3.7, introduces users to loops as an alternative to repeated sequential statements. This is similar to the previous challenge, but we do not put all the blocks on the canvas because we want users to learn about loops and how they simplify a series of repeated steps. As such, users are presented with two categories, Directions and Loops, from which they can select blocks in order to help Ada draw a square, as in the previous challenge. If a user selects the wrong blocks, for example, she picks a loop and a move forward block, then she can run the program, and from the robot's action, see that she completed the challenge incorrectly. The user is then able to reset the challenge and try it again. Users are also able to slow down the speed the robot moves at, so users can run the program and see exactly what movement corresponds to each block, enabling them to better understand the relationship between each code block and the statement being executed.

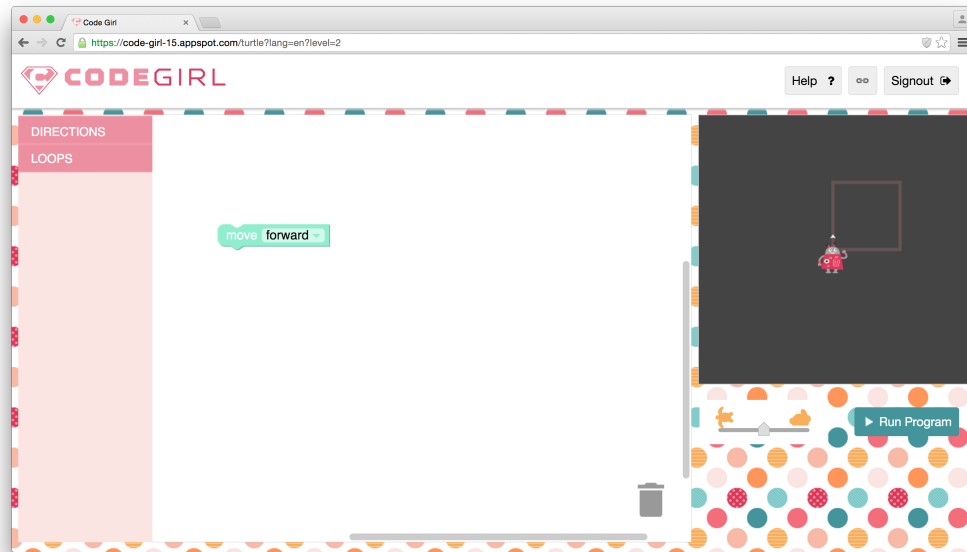


Figure 3.7: Users learn how to use loops in place of iteration to repeat actions.

3.3.4 Challenge 4: If-Else Conditional Statement

In the fourth challenge, seen in Figure 3.8, users are introduced to the concept of an if-else statement, an important programming paradigm. Users are tasked with the challenging of helping Grace rescue Ada, continuing the superhero story of the game. To complete the challenge, users must move Grace in one direction around a barrier to obtain a magic wand, and then change her direction to move toward Ada on the other side of a barrier. Thus, the if-else statement users must complete can be thought of as follows: “If Grace does not have the magic wand, move in one direction, otherwise, move in another direction.” Users are also introduced to angles, as they must select the correct angle to make Grace fly at to move around the barrier separating Grace and Ada. Angles are more advanced mathematical concept than is currently taught to children between the ages of five and eight, so the concept is abstracted slightly and users can use a popup dial, show in Figure 3.9, to select the direction for Grace to fly. As such, users need not be familiar with angles to complete the challenge, but in performing the task, are introduced to both this new math concept and the notion of an if-else statement.

3.3.5 Challenge 5: Do-Until Conditional Statement

In the fifth challenge, shown in Figure 3.10, users are introduced to a new type of conditional statement, a do-until statement. Users are tasked with navigating a person through a simple maze, which they can accomplish most efficiently by using a repeat-until block to repeat a step forward until the end of the maze is reached. All of the necessary blocks are given to the user from the start, so they can easily drag and drop different blocks

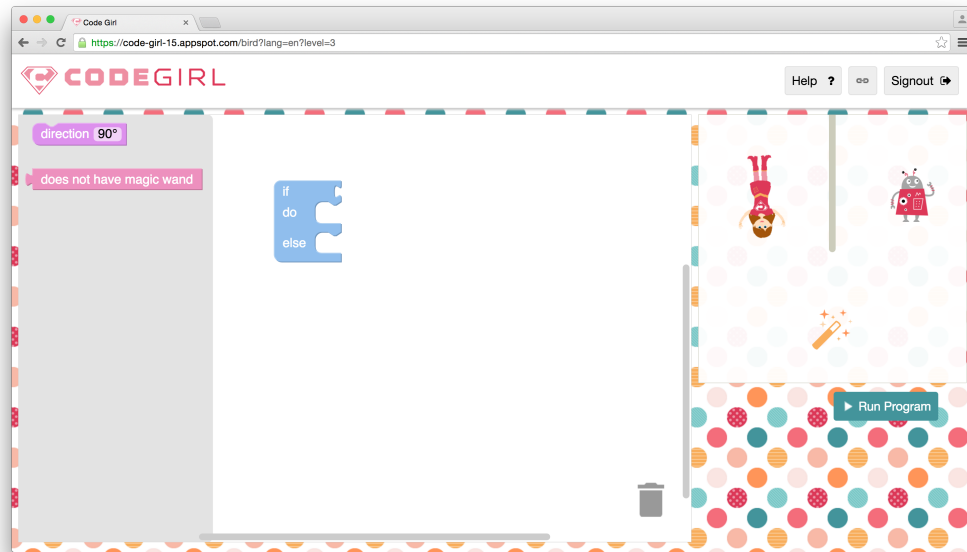


Figure 3.8: Users learn how to use an if-else statement.



Figure 3.9: Users are introduced to angles, abstracted as a dial.

to the Blockly canvas and experiment with using the until block to repeat actions until they feel they have successfully completed the challenge.

3.3.6 Challenge 6: Nested Conditional Statements

The sixth challenge integrates the lessons taught in the previous two challenges with another maze problem designed to introduce users to nested conditional statements. Users must once again navigate through a maze, but this time are presented with both a repeat-until block and an if block, the latter of which can be nested in the former to solve the maze problem with the minimum number of blocks, as shown in Figure 3.11. Users are not required to solve the maze using a nested conditional statement, but are allowed to explore this option as the most efficient solution to the challenge, giving users the opportunity to learn and explore these logic and computer science concepts at their own pace.

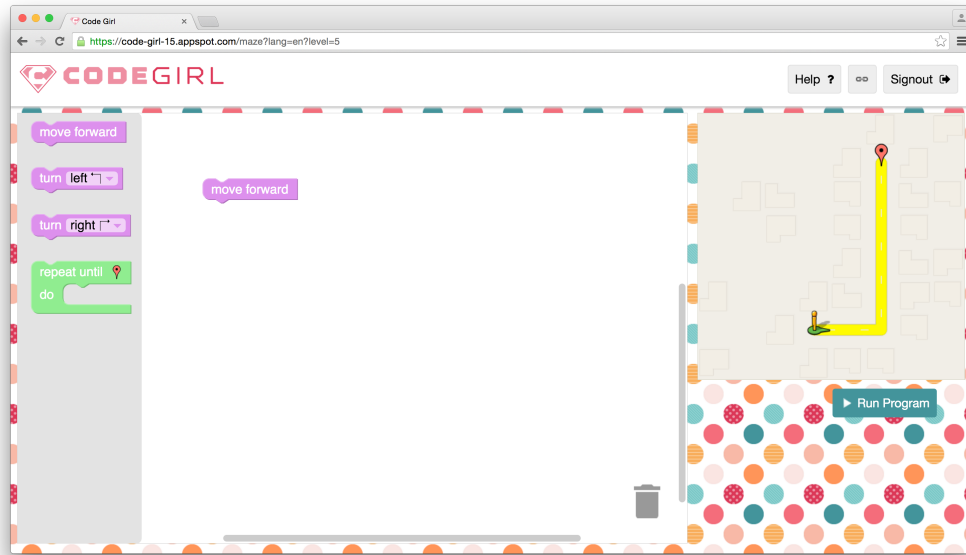


Figure 3.10: Users are prompted to use a do-until block to navigate through a maze.

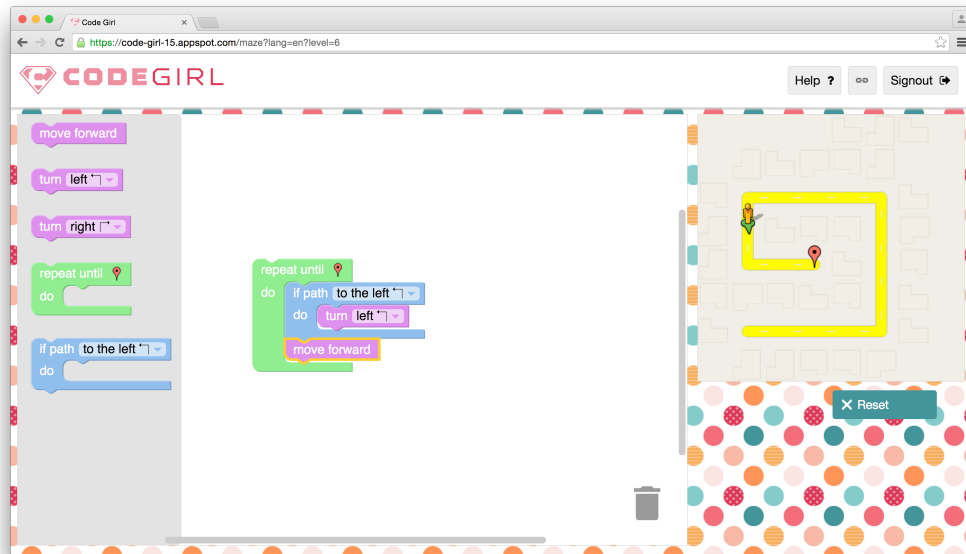


Figure 3.11: Users can learn to use nested conditional statements to solve a problem.

3.3.7 Challenge 7: Order and Conditions

In the seventh and final challenge of the game, depicted in Figure 3.12, users are instructed to help Grace dress for a superhero mission, a problem designed to allow users to demonstrate and test the concepts of order and conditions taught in previous challenges. Users are presented with two categories from which they can select blocks to dress up Grace. The blocks can be put together in many different ways; however, logical

conditions for the order of clothes must be met. For example, a user cannot put on Grace’s belt before she puts on Grace’s skirt or shirt. This challenge builds upon previous lessons taught, while also reinforcing basic logic concepts and maintaining the superhero story of the game, unifying the challenges before the final avatar customization level.

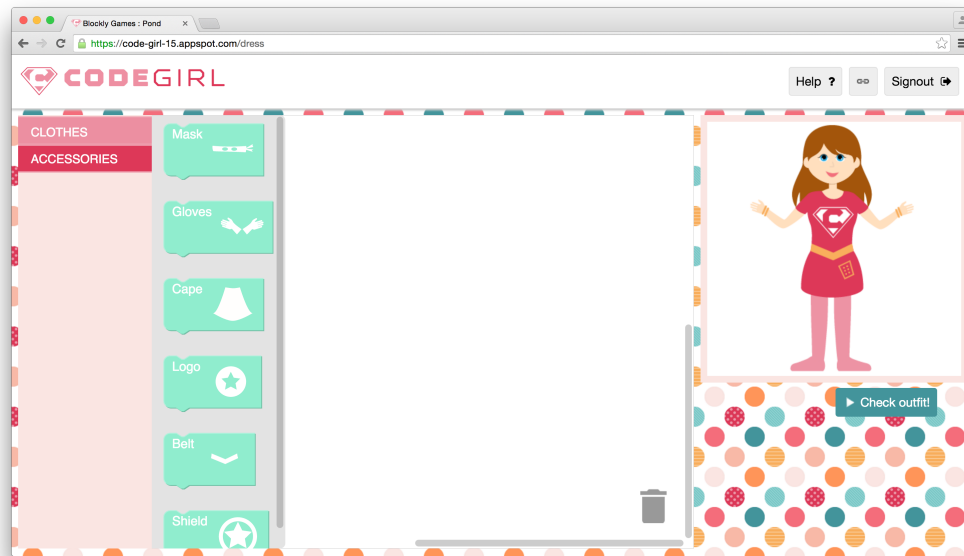


Figure 3.12: Users must dress Grace for a mission using the provided blocks.

3.4 Saving the Avatar

Once users are happy with how their avatar looks, they are able to save a picture of their avatar by selecting the “Save Image” button on the final avatar customization level, shown in Figure 3.13, and then print it out to share with their friends. The avatar is saved by converting the Visualization Box, represented with a canvas element, into a PNG image using the *html2canvas* script [28] and built-in `toDataURL()` Javascript method. The image is immediately downloaded with *download.js* if the application is running on Chrome or Firefox, otherwise, a new tab is opened with the final avatar image [30]. When users are done playing the game, they can log out at an avatar customization level, and the application will save their progress. Users can then return to this point in the game later on. If users want to create a new avatar, they can start the game over by clicking the “Start Over” button after saving their avatar.

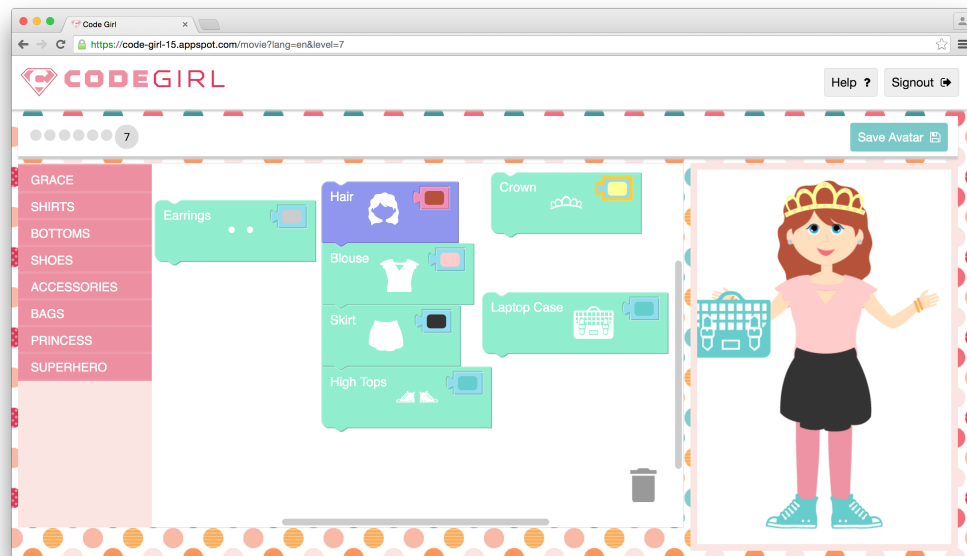


Figure 3.13: Users can save the avatar as image once done customizing.

Chapter 4

Project Management

4.1 Test Plan

Our test plan can be broken down into three main sections: testing of the avatar, testing of the games and testing both together. The first part involved unit testing on the processes of signing up, signing in, and using the Blockly pieces. We then conducted functional testing on the different combinations of Blockly pieces and the avatars they produce as we wrote the code to make sure that adding new features did not break existing code. Next, each of the challenges was tested separately. This phase tested that the correct inputs or combination of blocks produced the correct outputs, and therefore solved the maze or puzzle. Finally, we integrated the games with the creation of the avatar to test the system as a whole. An important part of this phase included testing the possible combinations of challenge outcomes, such as successes and failures, and the resulting actions, such as features unlocked. After we tested the system as the developers, we conducted both beta testing after the first year of development, and a formalized user study, as described in Chapter 5, after two years of development. Unit testing and functional testing were conducted in between beta testing and the user study, as features and challenges were modified and added to the game.

4.2 Project Risks

A risk analysis is conducted to identify possible risks and the costs they could have on the system or project as a whole. As shown in Table 4.1, each risk is given a probability from zero to one as well as a severity from zero to ten. The total impact of the risk is then calculated by multiplying the probability and severity together. By considering these risks at the beginning of the project, we aimed to take action to prevent the risks and their associated consequences from happening. However, the biggest risk we encountered was not one we included in our original risk analysis. Originally, we chose PHP as our language to communicate with the server because Google App Engine is compatible with PHP, and it is the language we know best. About three quarters of the way through the project, we realized that Blockly uses Python to communicate with Google

Risk	Consequences	Probability (P)	Severity (S)	Impact (P x S)	Mitigation Steps
Bugs and Errors	We will have to go back into the code, find out what's causing the problems, and fix them	1.0	3	3	Do ample testing every step of the way to find bugs sooner and use debugging software
We run out of time	Not all desired functions of the project are completed by the design conference.	0.3	9	2.7	Prioritize our work and implement the most important functionality first and perform extensive time management
Incomplete requirements	We do not implement a functionality our system needs	0.2	8	1.6	Review all our requirements to make sure they meet and match our intended outcomes and continuously talk to our advisor regarding the requirements
Changing scope/objectives	We might need to redo sections of code and it can delay our schedule	0.2	5	1	Prioritize requirements and functions to keep the project within the scope and limit the number of functions added after we begin implementation

Table 4.1: Our risk table for the project.

App Engine for storage of the user's blocks [17]. In order to successfully reload the users blocks and level when they log in, we would also have to use Python, a language none of us knew. The severity of the problem seemed pretty high, as one of us had to learn a new language to complete the project, and we did not know how easy or quick that would be. We were, however, able to do some tutorials and finish the required Python parts of our project to successfully integrate with Blockly. We learned that we should have included a risk about the technology or languages changing, so we would have been more prepared to handle the problem we encountered.

4.3 Societal Issues

As engineers, it is important to consider how the applications we create affect the world around us. We must conduct ourselves in a way that aligns with our ethics as well as the society as a whole. The following societal issues are discussed as they relate to us and the project *Code Girl*.

4.3.1 Ethics

Because we were the people defining the requirements and specifications for our project, we had to think differently about ethical issues in ways that a group creating something for an outside customer might not. For example, we did not have a company's code of ethics to follow, so we had to decide on our own. Also, unlike most other groups, we had to consider ethics involving children, since we developed our application for a certain age group and tested it with real children. These differences prove we had to carefully consider the ethics of our project.

Ethics in Our Team

Given the nature of our project, we examined the Association for Computing Machinery's (ACM) Code of Ethics and Software Engineering Code of Ethics [20]. The general ACM Code of Ethics talks about morals, professional responsibilities and organizational leadership. These helped us to prevent any ethical issues within our group and foster communication by driving us to be trustworthy and honoring assigned responsibilities. The Software Engineering Code of Ethics can be applied to how we as individuals act as developers. This code focuses mainly on acting consistently with the public interest and upholding integrity. We kept both of these in the front of our minds throughout our senior design project.

Ethics in Our Project

The most important ethical discussion we needed to have involved working with and designing for young girls. We wanted the web application to purposefully appeal to girls, however there could be sexism involved in us deciding what exactly that means. Obviously, not every girl likes pink and princesses, so it was important to offer a wide variety of clothing and accessories for the avatar that will appeal to a multitude of girls. Another ethical dilemma involves internet security because parents need to have control over what their children see on the internet. We had to take this into consideration if we wanted the child to have a profile where she can sign in and save her progress. There are legalities that go along with asking children to provide an e-mail address and password to sign up, usually a parent must do it for them. Lastly, we wanted to interact with young girls to test our project throughout the design process. We hoped to find out what appeals to them, ask them questions about our design and get feedback on how to make it better. There are legal issues with

contacting minors, as well as a Santa Clara University “human subject review process,” so we followed these procedures with conducting user testing. Making sure we uphold our ethics required some research to make sure we follow laws, make our interactions constructive, and make our web application safe and appropriate.

4.3.2 Social Context

From a social standpoint, we aim to interest a very specific group of young girls in computing to help benefit society. Although an interest in playing our game or learning to program does not necessarily mean the user will go on to study computing or have a job in the field, we hope to at least help spark the idea and the possibility at a young age. Computing is growing so fast that there are not enough graduates to fill all the jobs, or enough women in the jobs currently available, exacerbating the gender divide in technology. Diversity in computer science is lacking, an issue we further addressed by allowing users to customize Grace and change her skin, eye, and hair color and hair style. By promoting diversity with Grace, we aim to inspire and empower all young girls so we can fill the growing amount of computer science jobs with capable, diverse, smart women when they grow up.

4.3.3 Usability

User testing helped put us in the mind-set of five to eight year old girls to help us make *Code Girl* even easier and more intuitive to use. We included animations in the directions to help the young girls better understand how to use our application because their reading levels and vocabulary vary. The user can also slow down the animations in trying to solve the challenges to easily understand and learn what the blocks are doing. Usability is covered more in Chapter 5.

4.3.4 Lifelong Learning

Code Girl has prepared us for a life of learning new technologies as well as fueled a passion in us to help awaken the love of learning in new generations of young girls and programmers. This project has taught us how to quickly learn important and relevant parts of a language on our own. For example, we had to learn Python in a short amount of time to be able to integrate with Blockly and Google Cloud Storage to save what level the user was on. Learning how to distinguish what features of a language are applicable to the functionality we needed to implement proved a useful skill that will help us easily pick up new languages on our own in the future. *Code Girl* also inspired us to extend our passion for encouraging and supporting women in computing to a new generation. Channeling this passion, we want our program to reach as many young girls as possible, and we would love to partner with summer camps, schools and/or the Girl Scouts to include *Code Girl* as part of a curriculum to inspire young girls to learn how to program.

4.3.5 Compassion

Our compassion is evident in our desire to educate and enrich an often overlooked portion of the youth, who are not given many educational programs or games specifically designed for them. By beginning to teach basic computer skills and computer science ideas to young girls, we are giving them the opportunity to develop their interests and skills to help them succeed.

Chapter 5

Usability Study

A study was conducted to assess the usability of the application *Code Girl* and to receive feedback that could be used to improve the application. From the study, we hoped to learn how usable *Code Girl* is and to uncover both how the application currently meets research goals and how it can be improved to better meet the objective of interesting girls ages five to eight in computer science. Sections 5.1 to 5.3 provide a background on the motivation for and type of study conducted, including the study participants. Sections 5.4 to 5.5 cover how the study was conducted. Sections 5.6 to ?? provide a discussion of the usability measures evaluated and present both the results gathered, as well as an analysis of these results as they pertain to research goals.

5.1 Background

Game-based learning through applications such as Scratch and Alice has been proven effective in teaching children computer science concepts and skills [21]. Bell and Gibson identify three criteria that determine if an application will be engaging to children: active/passive, flow, and longevity [21]. Researchers also recommend creating girls-only groups for activities or interactions typically dominated by males, such as gaming or computing, to better engage girls and effectively reduce the gender divide that exists in these areas [24]. The study was thus designed to assess if *Code Girl* met these criteria by actively engaging users in the learning process, maintaining their interest in the activity through the use of story-telling, providing sufficient challenges and accessories to keep users playing, and appealing to the girls interests and perspectives.

5.2 Type of Usability Study

To assess whether *Code Girl* is effective at introducing and engaging users in computer science topics and skills and is usable by our target age group, we chose to conduct a usability evaluation study, followed by a focus group.

Usability evaluations give insight into users behaviors when presented with the application, and focus

groups give insight into users perceptions and attitudes toward the application, providing both qualitative and quantitative data about *Code Girls* usability. Both can be also be used to identify ways the application can be improved. We thus chose to conduct a summative usability test or lab study, which is done near or at the end of a products development, followed by a brief usability survey and then focus group, after consulting User Experience Research method guides to determine which methods would be most appropriate for the age range of our participants [25, 26]. Target users are between the ages of approximately five to eight, putting them in the Intuitive Thought, and perhaps Concrete Operations, stages of cognitive development, in which user testing and small focus groups are recommended research methods [25]. Additionally, due to the young age of the participants, the study was designed to use vocabulary suitable to their level of education.

5.3 Subject Population

Subjects were identified and recruited using the user profile we identified before designing and developing the application. We initially attempted to recruit twenty participants to complete the study, but due to time and resource constraints selected six participants, split over two study sessions. The participants were females, between the ages of approximately five to eight. Due to geographic constraints, participants were from the surrounding area and thus lived in Silicon Valley. Participants were recruited using the Recruitment Flyer, in Appendix A, and parents were provided with a Parental Consent Form, approved by the Santa Clara University Institutional Review Board (IRB) when their child was selected. Participants received a small gift as an incentive for participating in the study.

5.3.1 Subject Rights

The parents or guardians of participants were informed of procedures, intent of the study, and potential risks via an informed consent form, also approved by the Santa Clara University IRB to maintain ethical standards. Participants were made of aware of these things, as well as their right to withdraw at any time without penalty, during a verbal assent process and introduction to the study, before the usability evaluation and focus groups were conducted.

5.3.2 Subject Risks/Discomforts

The study posed minimal physical and psychological risks to participants. The risk and discomforts identified are: (1) stress, (2) breach of confidentiality, and (3) fatigue. Attempts to mitigate the above risks and discomforts were made before the study was conducted.

Stress

Psychologically, participants might have experienced stress during the usability evaluation, as they were told to complete a task within a limited time frame. The probability of the time limit causing stress could have been high, but was minimized to reduce harm by telling participants that although the goal was to complete the task in a certain time, they were encouraged to play at their own pace and ask questions if they needed to. Participants might have also experienced stress or similar discomfort during the focus group when prompted to share their thoughts and ideas if they were shy or self-conscious. The probability of this discomfort was less and the magnitude of this potential harm was not significant because moderator was instructed to foster an open, judgment-free environment for the group discussion, and participants were informed that they could always refuse to answer a question if they wanted to, without fear of reproach or needing to explain themselves.

Breach of Confidentiality

Even though the *Code Girl* application requires a user to create an account and log in with Gmail to access the games, which could be a minor breach of confidentiality issue if the e-mail could be associated with the participant, the probability of this occurring was determined to be zero because all of the participants were signed in to the application using a test account. Measures were also taken to maintain the confidentiality of the data collected from the study and forms.

Fatigue

The duration of the study, as explained below, is a lot of time for younger participants. This discomfort was mediated as follows. Snacks and refreshments were provided both during user testing and during the focus group for participants to enjoy if they were feeling fatigued at all during the study. Participants were told that they could step away for a snack or a beverage if they need a break. There was also a break in between the usability evaluation portion of the study and the focus group portion of the study, so the participants could move around and relax while the focus group was being set up.

5.3.3 Procedures to Maintain Confidentiality

Steps were taken to protect the privacy of participants and maintain the confidentiality of data gathered before, during, and after the usability study. For more information, see the IRB Application for this study [33]. Participants, furthermore, were not asked to give permission for release of identifiable data at the time of the study or in the future because the audio and video recordings were only used for collecting data anonymously. The data collected from the usability evaluations, the follow-up survey, and the focus group recordings was

Table 5.1: Study Timing

Activity	Timing
Introduction	5 min.
Usability Evaluation and Survey	
Lab Study (Game Play)	45 min.
Usability Survey	15 min.
Focus Group	
Introduction	5 min.
Key Topic 1	10 min.
Key Topic 2	10 min.
Key Topic 3	10 min.
Wrap-up	5 min.
What's missing	5 min.
Conclusion	5 min.

not coded.

5.4 Location and Time

The user study, which spanned two sessions, was conducted in the morning, on the Santa Clara University campus, and was designed last approximately two hours. The session were conducted in a room contained several desktop Mac computer stations, as well as projector screens, which the Focus Group questions were displayed on. Mac computers were chosen so participants interaction with the application during the usability evaluation could be recorded using QuickTime Players Screen Recording. The evaluations were lead by the assistant researcher and supervised by the primary investigator.

5.5 Structure

A summary of the structure of the study, including the main activities and approximate timing for these activities is shown in Table 5.1.

The structure of the study was tested via a pilot study with one participant. Following the pilot study, the usability evaluation survey questions were modified slightly and timing of the activities was set as shown in Table 5.1. No changes to the focus group were made.

5.5.1 Introduction

The complete script for the usability evaluation introduction is provided in Appendix B. Before the study began, participants were asked to supply the informed consent forms signed by their parent or guardian. Participants that did not supply this form were removed from the study.

The two investigators introduced themselves before welcoming the participants and thanking them for their participation. The participants were informed of their rights and the purpose of the study and their assent was obtained via the verbal assent form.

5.5.2 Usability Evaluation

The participants were then introduced to *Code Girl*. They were instructed to begin using the application and play until they have reached level 3 of the application. They were given 45 minutes to complete this task, but encouraged to continue playing if they finished the task before the time was up. Participants were informed that they could ask questions if they did not know what a word means, if the instructions were unclear, or if they needed assistance completing a task. Questions were noted, but not associated with participants. Participants time on each level and challenge was measured, as well as the number of errors they made.

After 45 minutes were up, all participants were given the anonymous usability evaluation survey and given 15 minutes to fill out the survey, included in Appendix C.

5.5.3 Focus Group

An investigator then conducted a focus group, which was designed to last approximately 1 hour, but in both sessions lasted less than twenty minutes. Each focus group was recorded using QuickTime Players Audio Recording.

The investigator gave the predefined introduction, in Appendix D, and then progressed through the list of questions, using a slideshow of the questions as a supplement so participants could refer back to the questions during the course of the discussion. Group discussion among the participants was encouraged. The investigator acted as a moderator, prompting participants to share their opinions and feelings, managing the time, making sure some participants do not dominate the conversation, and asking for further explanation or clarification if necessary.

5.5.4 Conclusion

Participants were thanked for their participation and encouraged to contact the investigators if they had further questions or would like to know the results of the study. The participants were then given a small gift for their participation.

5.6 Evaluation

We analyzed both performance metrics and self-reported measures to assess how *Code Girl* meets the standard usability goals of efficiency, effectiveness and satisfaction, and determine if our application meets our

goal of interesting and educating girls in computer science.

5.6.1 Performance Metrics

Typical performance metrics include: (1) task success, (2) time on task, (3) errors, (4) efficiency, and (5) learnability. Given the nature of our study, we decided to measure the following performance metrics:

1. **Task Success:** The task can be either a success or failure. We defined a successful task as the one in which the participant was able to reach Level 3, with or without assistance.
2. **Time On Task:** We measured the time on task for the task using the video recordings. It was calculated as the difference between the time at which the participant either completed the task or was told to stop because time was up and the time at which the participant clicked on the “OK” button in the dialog box for the instructions in the first challenge to begin playing the game. Time on task is an important measurement because it gives an indication of which challenges and levels are the most difficult, and may in the extreme case be so difficult that users may give up or quit the application and not return.
3. **Efficiency:** We measured the efficiency of the application as the overall relative efficiency of system, which is calculated by dividing the time taken by the users who successfully completed the task by total time taken by all users, as follows, where N is the number of tasks, R is the total number of users, and t is the time taken to complete the task: [32]

$$OverallRelativeEfficiency = \frac{\sum_{j=1}^R \sum_{i=1}^N n_{ij} t_{ij}}{\sum_{j=1}^R \sum_{i=1}^N t_{ij}} * 100\% \quad (5.1)$$

Errors were not counted, though questions were noted and analyzed for potential usability issues, as children learn from asking questions and through hand-on experience. For this system, hands-on experience includes putting blocks in the incorrect order and making other mistakes, which they are then prompted to correct. Errors should help users learn in this game, and thus not be counted as a usability issue.

Additionally, learnability, though it seems like it would be a useful measure for assessing the usability of an educational game, was not a relevant measure for studying this application, as the goal is to determine whether users learn concepts from the application and not learn how to use the game. *Code Girl* was designed to be intuitive and teach users as they play at their own pace, with the assumption that some users may have never been exposed to a visual programming environment and thus need longer to play around and learn how to interact with the game. Learnability, furthermore, is typically assessed over time via multiple studies with the same participants, which was not feasible given time constraints for developing the application.

5.6.2 Self-reported Measures

We collected self-reported measures via the usability survey and the focus group to assess participants satisfaction with our application and to determine ways the application could be improved.

Usability Survey

The survey consists of questions covering three main categories: (1) participant background, (2) prior experience with programming games, and (3) perception of the application. The categories were chosen so research goals could be evaluated with respect to factors such as the user's age and prior exposure to computer science concepts or games. Perception of the application was assessed via questions loosely based on a System Usability Scale (SUS) Survey, but modified to address specific features of the application and the young age of users [34]. Additionally, it is recommended that rating systems incorporate visual scales instead of numbers or words when working with elementary school children, so the usability evaluation survey uses visual icons for the Likert-type scale questions so participants could more easily understand and answer them [26].

Focus Group

For the focus group, questions that matched our topic and objectives were identified. The final questions, which include the four key topics and wrap-up, were then selected and refined based on which would best provide insight into how users felt about the current application and how they think the application can be improved. A summary of the topics, the questions, and the goals for these questions is shown in Table 5.2.

Participants responses to these questions were recorded, and then later reviewed and categorized according to the goals for the question to primarily assess user satisfaction and specific ways the application can be improved.

5.7 Results

The performance metrics and self-reported measures collected from both sessions of the usability study are presented below. All usability evaluations were performed with the application running on Mac desktop computers with Chrome as the browser.

5.7.1 Performance Metrics

Task success and time on task were measured from the video recordings of the usability evaluation, with efficiently calculated based on the observed values.

Table 5.2: Focus Group Structure

Topic	Question	Goal
Introduction	Please introduce yourselves, telling us your name and what your favorite game is	Start participants talking and make them comfortable
Key Topic 1: General perception of the application	What did you like most about <i>Code Girl</i> ? What did you like least about <i>Code Girl</i> ?	Assess if users in general like our product and identify specifically what they do or not like about it
Key Topic 2: Barriers to using the application as intended	What problems or challenges did you run in to?	Identify issues preventing users from enjoying or benefiting from the application
Key Topic 3: Wants and Needs	What would you add to <i>Code Girl</i> ? What would you remove from <i>Code Girl</i> ?	Learn how we can better meet users wants or needs
Wrap-up	Did you learn anything playing the game? If so, what did you learn?	Assess if we are meeting our goal of educating and inspiring users
What's missing	Any other comments?	Let participants express any thoughts or ideas they care about but were not mentioned

Table 5.3: Usability Evaluation Results

	Participant	Task Success	Time on Task	Level Reached	Total Time Taken
Session 1	1	Yes	28 min, 58 sec	Avatar 4	45 min
	2	Yes	23 min, 7 sec	Challenge 5	45 min
	3	Yes	24 min, 14 sec	Challenge 5	45 min
Session 2	1	Yes	8 min, 21 sec	All Completed	23 min, 40 sec
	2	Yes	21 min	Challenge 6	45 min
	3	Yes	14 min, 14 sec	All Completed	41 min, 6 sec

Table 5.4: Average Time on Task

	Mean Time on Task
Session 1	25 min, 26 sec
Session 2	14 min, 32 sec
Total	19 min, 59 sec

Task Success

All participants in both sessions successfully completed the task of reaching the third avatar customization level in the 45 minutes allotted for the evaluation. In fact, two participants completed the entire game in the given time. Table 5.3 summarizes the results of the usability evaluation, including task success, time on task, level reached after the allotted time, and total time taken.

Time on Task

The time each user took to complete the given task is shown in Table 5.3. The maximum time given to participants to complete the task was 45 minutes, but no participant took the maximum amount of time. The average time on task across both sessions was 19 minutes and 59 seconds, with a mean of 25 minutes and 26 seconds in the first session and a mean of 14 minutes and 32 seconds in the second session, as summarized in Table 5.4.

Task Efficiency

Since all participants successfully completed the task in the given time, the numerator in Equation 5.1 is reduced to the sum of the time each user spent completing the given task. Using the data from Table 5.3, the total time spent by all users on the task across both user sessions was 19 minutes and 59 seconds, yielding an efficiency of 100%, as summarized in Table 5.5.

To get a more nuanced view of user efficiency, efficiency for the first three challenges and the first two

Table 5.5: Efficiency

Task	Task Completion Rate	Average Time on Task	Efficiency
Reach Level 3	100%	19 min, 59 sec	100%

Table 5.6: Sub-task Efficiency

Sub-task	Average Time on Sub-task	Efficiency
Challenge 1	2.49 min	40%
Avatar Level 1	4.46 min	22%
Challenge 2	6.28 min	16%
Avatar Level 2	1.44 min	70%
Challenge 3	5.33 min	19%

avatar customization levels encompassed in the given task was calculated. Efficiency was calculated by obtaining the ratio of sub-task success rate and mean time on sub-task, with the sub-task being the challenge or customization level completed. Since all participants completed the task, all of the sub-tasks have a success rate of 100%, reducing the efficiency to the inverse of the mean time on each sub-task, or the mean time on each challenge or avatar customization level up to the third avatar customization level. The efficiency of the challenges as compared to the avatar customization levels is shown in Table 5.6. As can be seen in Table 5.6, participants were least efficient with the two challenges designed to introduce them to programming concepts and were actually most efficient with the initial challenge designed to introduce them to the Blockly environment and the second avatar customization level. Participants, however, will still more efficient on average with the first avatar customization level than they were with both the second and third challenges.

Requests for help

Questions and other requests for help posed during the usability evaluation were documented instead of errors, and are presented below with the context in which they arose, with duplicates removed:

- I'm stuck - In a challenge
- How do I get rid of this? - Throwing a block in the trash can in an avatar customization level
- How do I start? - Closing an instruction dialog box in order to begin playing the game
- What am I supposed to do? - Confusion regarding how to complete Challenge 2
- How do I make her move this way? - Confusion about the angle dial in Challenge 4

- What do I do now?/ How do I play more? - Unlocking a challenge to get more accessories and clothes
- What does this mean? - Lack of an error message in Challenge 7 when not all of the blocks are present on the canvas
- What is that word? - Did not know what one of the words in a dialog box meant
- How do I move this? - Using a mouse to drag and drop blocks tricky
- What do I do with this block? - Regarding the nested conditional blocks

Participants in the first session asked more questions than participants in the second session, with participants in the first sessions frequently requesting help reading words and progressing through a challenge. Participants in the second session asked fewer questions about how to progress through a challenge, except for with Challenge 4. All participants asked questions about how to use the angle dial to make Grace fly in a certain direction. Though they did not often ask questions about this, participants in the first session expressed frustration at using the mouse to drag and drop blocks onto the Blockly canvas. Participants in the second session did not ask questions about how to move blocks around except for when it came to removing blocks from the canvas. All but one participant asked a question about how to remove a block in either a challenge or in one of the avatar customization levels.

5.7.2 Self-Reported Measures

Data was collected from the usability survey and focus group to further assess *Code Girl's* usability, in particular, user satisfaction with the game, and how well the application met research goals.

Usability Survey

The complete survey results are provided in Appendix E. Regarding the first two categories of survey, which include participant background information and prior experience with programming games, the following data was collected. Participants were between the ages of six and nine. All participants indicated that they had programmed or coded before, with “Scratch” selected three times, “Code.org” selected twice, and “Alice” and “Other” selected once each as applications that they had used before.

Self-reported usability measures, derived from the results of the third category of survey, which focused on user perception of the application, are presented in Figure 5.1. Note that unless otherwise stated, agree should be interpreted as the responses “Agree” and “Strongly Agree.” Likewise, disagree should be interpreted as the responses “Disagree” and “Strongly Disagree.” All participants agreed that they liked the clothes and accessories available in the avatar customization levels and that they liked dressing up Grace.

There was the most variability in responses to the seventh question regarding the difficulty of the challenges, with a third of participants disagreeing that the challenges were hard, a third of participants agreeing that the challenges were hard, and a third of participants giving a neutral response. All but one participant strongly agreed that they are interested in programming after the game, and only one disagreed with this statement, though she did not strongly disagree with it. Likewise, all but one participant agreed that they would play *Code Girl* again, with the one participant who did not agree with the statement writing “I don’t know,” which was interpreted as a neutral response.

In the free-response box for additional comments, one participant responded “I think it was a great game. I liked all the things.” No other comments about the game were provided.

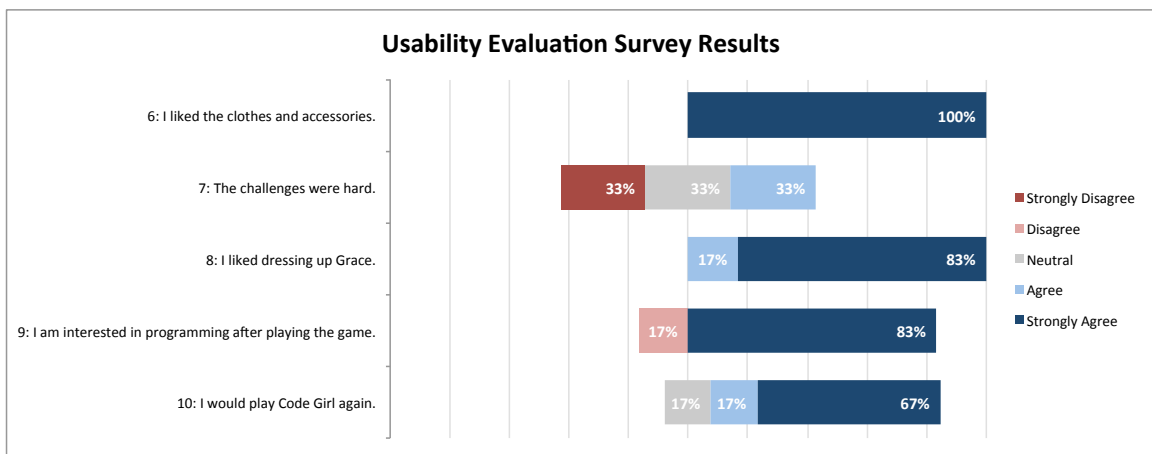


Figure 5.1: User perception of the application, across both sessions

Focus Group

Responses to each research question posed during the Focus Group for both sessions are summarized by session.

1. *What did you like most about Code Girl? What did you like least about Code Girl?*

In the first session, each participant responded differently, with one participant stating that the part of the game she liked the most was “dressing up Grace.” Another participant asserted that her favorite part of the game was “trying to go through the mazes,” whereas one participant stated that they liked the mazes least. The third participant stated that she liked “looking at [Grace]” and did not dislike anything about the game. One participant stated that her least favorite part of the game was “getting the answers wrong.”

In the second session, one participant adamantly stated that she “liked dressing up Grace,” and another agreed with her. The third stated that she liked that you had “to complete a level to get new accessories.”

When prompted to share what they liked least about the application, one participant asserted that she “liked everything about the game” and another indicated that she did not dislike anything about the game.

2. *What problems or challenges did you run in to?*

In the first session, one participant referenced the second and third challenges, adamantly stating that she had problems drawing the square. When prompted to explain what would have made it easier for her to complete these challenges, however, she did not provide a response. One participant asserted that the directions were “hard to understand,” and another stated that she could not read some of them because she did not know certain words, but both agreed that the visual instructions were good.

In the second session, one participant stated that she did not encounter any challenges or problems in completing the game. Another participant indicated that she found the sixth challenge to be problematic, as she attempted to complete the maze without using nested conditional statement blocks. The third participant indicated that she encountered problems with the fourth challenge, and another participant agreed. When prompted to explain what would have made the challenge easier for her to complete, she responded, “more blocks to tell you the direction, like go down...and then to the side,” and another participant again agreed with her.

3. *What would you add to Code Girl? What would you remove from Code Girl?*

In the first session, one participant responded that she would add “more easy stuff” to the game, like coloring, but another participant argued against this by stating that then the game “wouldn’t be fun.” One participant stated that she wanted more math in the game and would add more accessories. When prompted to describe the accessories that she would add, she mentioned “more earrings.” The third participant wanted more of a celebration when challenges were correctly solved. Additionally, one participant wanted to remove the second and third challenges.

In the second session, all the participants stated that they would add more accessories. When prompted to explain what accessories they would add, the participants listed background, objects to hold like a football, and career-focused outfits. One participant asserted that she would add a boy avatar that could be customized like Grace and a challenge where you bake a cake. The other two participants agreed, indicated that she would like “choices if you want to make a girl or a boy,” and suggested creating a series of cake-baking challenges where “every time you complete a level you get more ingredients.” Only one participant suggesting removing something from the application, and requested that the second maze challenge be removed.

4. *Did you learn anything playing the game? If so, what did you learn?*

In the first session, one participant claimed that she “learned nothing.” Another participant described how in the second challenge, there was a pattern for how the blocks were connected to successfully complete the challenge and draw a square, and the following challenge was “kinda the same, but it did less work because you could repeat it,” indicating an understanding of sequential code and loops. In reference to the fourth challenge, the same participant described how you could click on the number in the block and use the dial to change the number to make Grace move in the correct direction.

In the second session, one participant simply stated “Yes,” while another stated “No.” When prompted further, she responded that she learned “to memorize what to do,” which was clarified to mean she learned how to repeat certain actions to solve the challenges. The participant who responded “Yes,” when prompted further, indicated that she learned that blocks that repeat actions “take you less time, like on the second one where you have to do a square,” and the third participant agreed.

5. *Any other comments?*

In the first session, one participant responded that the game was fun and that she liked everything except for the mazes. One participant said that some of the instructions were hard to understand and when prompted to explain what would make the better, answered that it would be helpful “if you could just make up your own instructions.” One participant stated that the game was “too easy.” She and one other participant added that there should be more versions of the game or more challenges and levels so they could play it again or play for longer.

In the second session, one participant stated “I hope you do another thing,” requesting more, similar applications to play at home since she was able to complete the entire game in the allotted time. Another participant requested another version of the application designed to be a bit more challenging, and the former agreed that it was “too easy.”

5.8 Analysis

The identified performance metrics and self-reported measures were analyzed together to assess the usability of *Code Girl* and to determine how well the application met research goals, which included identifying ways to improve its short-comings in both these areas.

5.8.1 User Behavior during Usability Evaluation

Based on the requests for help received and participant behavior observed during the usability evaluation, it is likely that certain changes need to be made to the application to make it easier to use. These include more hints

to help users progress through challenges on their own and simplified instructions so users can more easily understand them and actually take the time to read them. Some of the words used in instructions were above the reading-level of some of the participants and the text was long enough that some participants did not read the directions, instead choosing to just start playing, though they would then often become confused about how to complete the challenge. The fact that all participants were confused by the angle dial in Challenge 5 indicates that this concept needs to be simplified as well, as it presents a barrier to successfully completing the challenge. Participants, however, did not ask many questions while on the avatar customization levels, suggesting that this portion of the game was engaging and relatively easy to understand and play. The one question that did arise on these levels was about how to remove blocks from the canvas, indicating that the trash-can functionality already present needs to be made more explicit and/or more intuitive.

Additionally, the desktop gaming environment is not very intuitive to young users, as many participants struggled to use the mouse to drag and drop blocks and/or asked questions about how to move and connect blocks on the canvas. As such, it is critical that the application be fully responsive to users can play the game on a browser using a tablet, an environment they are more familiar with. It may even be advisable to develop a native app for tablets, because even though we want to familiarize girls with using a computer, they may be discouraged from playing if they have to use an environment they are uncomfortable with.

5.8.2 Task Success, Time on Task and Efficiency

All participants completed the task of reaching the third avatar customization level in the allotted time, suggesting that there were not critical usability issues preventing users for playing the game. The time spent on the task and final level reached, however, varied significantly between the two usability study sessions. Two of the three participants in the second session performed the task substantially faster than the participants in the first study, and even finished the game before time was called. This could be because the participants in the second study were on average older than the participants in the first session, and thus could more easily understand the text-based instructions in the game. Another reason could be that even though the survey results indicate that all participants were familiar with programming before playing the game, participants in the first session expressed that they did not frequently play programming games such as Scratch, whereas participants in the second session expressed more familiarity with the games. Participants in the second session were thus more familiar with the concepts and visual programming environment of the game and able to complete tasks faster.

Additionally, all of the participants in the second session indicated that another person living in their home plays these programming games or programs, whereas only one person in the first session responded affirmatively to this question. This suggests that participants that have already been exposed to programming

via games and other people in their home are faster those less familiar with programming at playing the game, thus the game may be more usable to those already familiar with a visual programming environment and/or computer science. These findings were not surprising, as presumably users are faster at completing tasks they are already familiar with, and do not suggest a substantial usability issue, as the application was designed and developed for users to play at their own pace. However, more hints, some designed at familiarizing users more with the Blockly environment, could be provided to help those less familiar with computer science and programming games complete tasks efficiently and effectively, so they can learn without becoming discouraged.

The 100% efficiency rate was also not surprising, in that a time limit of 45 minutes was set for the usability evaluation under the assumption that users should be able to complete the task in that time with assistance. A less than 100% efficiency rate would likely indicate that the game was overly complicated, and thus likely to result in some users quitting and not returning to the game since they could not progress through levels and challenges as a reasonable rate. A high degree of variability in efficiency was also expected given that users have varied experience with other coding games and visual programming environments, with more experienced users able to complete levels and challenges more efficiently than those with less familiar with programming concepts and games.

The break-down of efficiency per sub-task, however, was surprising in that it indicated that users were not very efficient with the first avatar customization level, with an efficiency rate of 22% of the goal/minute completed, and were actually more efficient at the first challenge. This finding is likely attributable to the fact that users wanted to spend time customizing Grace and exploring all of the options available to them when they are first introduced to her, an explanation supported by their behavior, as many participants repeatedly changed Grace's physical appearance and shirts in the first level. The low efficiency rate for both the second and third challenge is not surprising, given that these challenges are designed to teach users concepts they are likely unfamiliar with, and thus will likely be difficult for them. When viewed in combination with the amount of questions asked by participants during these challenges and focus group responses, however, these efficiency results indicate that more hints likely need to be provided to help users complete these and later challenges a bit more efficiently, in order to prevent them from becoming frustrated or discouraged and potentially abandoning the game. One participant in the first focus group session repeatedly expressed dissatisfaction with the difficulty of the second and third challenges, making it clear that the difficulty of the challenges, with the exception of the first poses a moderate usability issue that should be addressed.

5.8.3 User Satisfaction

Based on both survey responses to questions “I liked the clothes and accessories” and “I liked dressing up Grace” as well as participants responses to the questions “What did you like most about *Code Girl*?” it can be concluded that all participants were satisfied with the avatar customization portion of the game. All participants agreed that they liked the clothes and accessories, as well as dressing up Grace, with most all of them strongly agreeing to these statements, and half of the participants across both focus group sessions responded that the part of *Code Girl* they liked most was dressing up Grace. Though some participants requested more clothes and accessories be added to the avatar customization levels, the general perception of this part of the game was positive and participants appeared satisfied with the customization experience and options available to them.

With regards to the challenge, we anticipated that the older participants would find the challenges easier to complete, and thus be more satisfied with them, than younger participants, who may be find the instructions and/or concepts being taught more difficult. However, analyzing the responses to the seventh survey question, shown in Figure 5.2, which was designed to assess user satisfaction with the challenges did not support this conclusion, as can be seen in Figure 5.2. It was assumed that participants who strongly agreed that the challenges were hard were likely dissatisfied with the experience of playing them, however, no participants fell into this category. Moreover, younger users were more likely to either respond neutrally to the question or disagree with the assertion that the challenges were hard, suggesting that they were satisfied with the level of difficulty of the challenges, perhaps more so than the older participants. Factoring in feedback from the focus group sessions, in which one participant requested two challenges be removed from the application, but two participants requested more levels and challenges, it was then thought that user satisfaction with the challenges was influenced by a factor other than age.

To investigate this further, participants responses to the same question were analyzed with respect to a different background criteria, their responses to the question, “Do other people in your house play these games or do computer programming?” Since participant age did not seem to correlate to their satisfaction with the difficulty of the challenges, it was then hypothesized that users who have been exposed to programming via someone close to them would be more familiar with or receptive to the concepts being taught and thus be more satisfied with challenges. Results, however, were similarly inconclusive.

During the focus group however, one participant indicated that she liked the challenges most in the game and asserted that adding simpler challenges would negatively impact the game and it “wouldn’t be fun,” indicating that the difficulty of the challenges may be appealing to some users, perhaps depending on their level of interest in logic and problem-solving. It could be that some users perceived the challenges to be

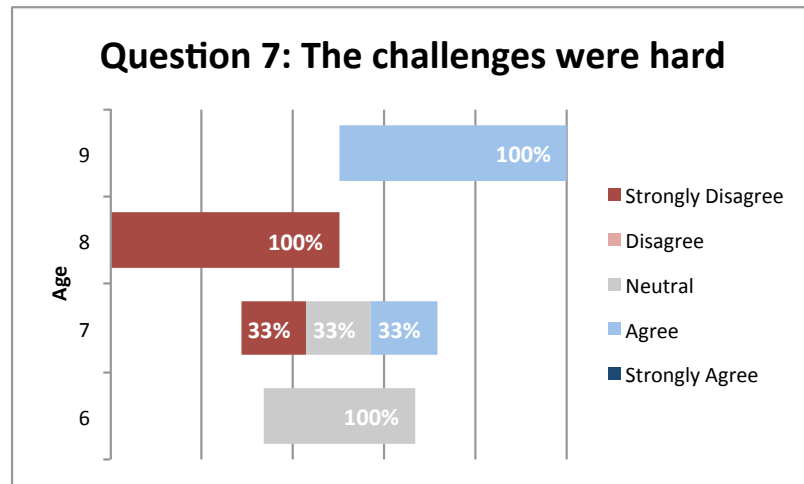


Figure 5.2: User perception of the difficulty of the challenges, by age

difficult, but still enjoyed the challenge of problem-solving, whereas others thought the challenges were either too easy or too hard and did not were not satisfied the problem-solving experience. The mixed results regarding user satisfaction with the challenges, however, are not very surprising or alarming in that it cannot be expected that all potential users will enjoy learning computer science.

Despite inconclusive results regarding user satisfaction with the challenges, overall participants appeared satisfied with the experience of playing the game. All but one participant agreed that they would play the game again, suggesting that they perceived the game to be engaging. Complaints about the game in general received during the focus group were mostly about the game being “too easy” to complete. Four participants requested more versions of the game, one even requesting a more challenging version, or more challenges and levels so they could play it again or play for longer. Their responses suggest that users may be dissatisfied with the duration and difficulty of the game, but only because they were satisfied with the overall experience of playing the game and would like to continue learning and dressing up Grace.

It should be noted, however, that all of the participants in the first focus group session expressed a degree of dissatisfaction with the instructions provided in the game. Participants explained that the instructions were difficult to understand and/or contained words that they were not familiar with, which made completing parts of the game, the challenges in particular, difficult. The girls did seem satisfied with the visual instructions, agreeing that these helped them understand how to complete tasks in the game, indicating that more visual or even audio instructions should be incorporated into the game to better accommodate user needs.

5.8.4 Research Goals

Primarily self-reported measures were used to assess whether or not *Code Girl* met research goals of interesting girls ages five to eight in computer science. All participants indicated having prior experience with

programming or coding games, thus *Code Girl's* efficacy in interesting someone in computer science who has never been exposed to the field before could not be analyzed. All but one participant, however, reported that they were interested in programming or coding both before and after playing the game, with the one who did not indicating that she was interesting in programming before playing the game, but not after. Upon further investigation, however, this participant strongly agreed that she liked dressing up Grace and strongly disagreed that the challenges were hard, which suggests that perhaps the challenges did not successfully appeal to her interests or capture her interest. Not all users, though, may be interested in programming even after playing the game no matter how engaging it is to other users, thus this result does not indicate a critical usability issue or failure to meet research goals, especially since all other participants indicated that they would play the game again and strongly agreed that they were interested in programming after playing the game.

Results regarding *Code Girl's* efficacy in teaching computer science concepts and skills are slightly less conclusive. When asked in the focus group if she learned anything when playing *Code Girl*, one participant in each session responded in the negative. However, when reminded of the challenges, she described an understanding of needing to repeat certain actions to solve some of the challenges. Two other participants described following a pattern to complete the second challenge and then using a simplified approach to achieve the same goal in the third challenge. One girl described this as “[doing] less work because you could repeat,” and the other observed that the second approach “[took] you less time,” demonstrating an understanding of the concept of loops simplifying the repetition of actions and using fewer steps to achieve the same result as sequential code. Other girls agreed that they learned the concepts mentioned by their fellow participants, suggesting that even though they may not initially be fully cognizant of the skills and lessons being taught, users are engaged in challenges and learning about computer science as they play.

Many participants described *Code Girl* as fun, and one responded on the survey, “I think it was a great game,” an indication that *Code Girl* successfully captures the interest of the target age group. One participant, furthermore, mentioned that she part of *Code Girl* she liked most was that you had “to complete a level to get new accessories,” suggesting that the application meets our goal of incentivizing game-play, teaching users computer science concepts, though they may not be explicitly aware of the lessons and skills being taught, while appealing to their interests of dressing and customizing a character like they do in other, non-educational games.

Chapter 6

Conclusion

The application, as initially designed and subsequently refined using feedback and analysis from testing, is available at <https://code-girl-15.appspot.com/>. The application is deployed and running on Google App Engine, with regular updates to address changes to the Blockly library and bug fixes. Additional changes, as described in this chapter, will be undertaken to further refine the application, though the provided steps can be taken now to release the application for widespread educational use.

6.1 Future Work

To better meet research goals of educating young girls and hopefully inspiring them to pursue a career in computer science, improvements based on data gathered from the usability study can be made to the application. As these changes are incorporated into *Code Girl*, the steps outlined below can be taken to integrate *Code Girl* into educational programs.

6.1.1 Application Improvements

Based on the results of user testing, improvements including both modifications and additions can be made to better meet research goals of educating and inspiring young girls.

Instructions

Younger users experienced more difficulty reading and following the instructions provided, thus instructions for both the avatar customization levels and challenges could also include audio instructions in addition to text and pictures. Following this idea, the instructions for first avatar customization level have already been changed to include a video tutorial demonstrating how to add blocks to the canvas, change the color of the items represented by the blocks, remove blocks from the canvas, and play a new challenge.

Challenges

Some participants also expressed that the second and third challenges, the ones in which the user is tasked with drawing a square using sequential code and then loops, were too difficult, suggesting that these challenges should either be simplified or removed from the application. Other participants, however, asserted that the game was too easy, and that more challenges should be included or a more difficult version of the application should be developed. To satisfy both users who thought the challenges were too difficult and who thought they were too easy, additional challenges that build on the concepts previously taught, and thus are more complicated, should be included in the game, but additional assistance should be provided so all users can complete them. All of the challenges should provide additional hints to help users who may need more feedback to successfully complete challenges by themselves, while ensuring that these users still learn the computer science concepts being taught. These hints can be based on the time a user has been on a challenge, prompting them to use a certain block if it appears that they have not taken an action in a long time, or provide them feedback if they have taken a wrong action and are now stuck in a challenge.

Including more celebration into the congratulations message that is displayed when users successfully complete a challenge, as was requested by one of the participants, may also increase user satisfaction with the challenges. This celebration could include music or a short video incorporating Grace and Ada to further the story-telling aspect of the game and further engage users in the game.

Avatar Customization

Although many additional accessories, such as the princess themed accessories, were developed in response to feedback from beta-testing, even more options can be included to enhance the customization portion of the game. As was mentioned in the focus groups, more clothes and accessories, such as those relating to careers like athlete or doctor could be developed to further inspire and empower users.

Additionally, diversity in the form of allowing users to change Grace's hair style as well as hair, eye and skin color were implemented in part as a result of feedback from beta testing. More diversity could be incorporated into the game by allowing users to customize a male avatar as well, as requested by two participants in the user study. This change, although it may appeal to the interests of some young girls, may also detract from the goal of creating a game that engages young girls in computer science by shifting the focus away from girls in technology. More research should be conducted with potential users before this change is implemented.

6.1.2 Integration into Educational Programs

To reach more users, and thereby introduce more girls to computer science, *Code Girl* can be integrated into local and national education programs. To incorporate the application into educational programs and initiatives, we can meet with local educators, such as elementary school teachers and principals, to discuss using the application as a supplement or addition to current curriculum. We can also reach out to leaders of local after-school programs and summer camps, some of which, like Girls who Code, operate at a national, or even global level [2]. The application only requires access to a computer or tablet and a modern browser, which is provided by many schools, either in class or via libraries and/or computer labs, and educational facilities, making it easy to integrate into educational programs.

6.2 Lessons Learned

There are few resources designed specifically for young girls that teach basic computer science skills. To fill this gap, we developed *Code Girl*, which uses Blockly to interest girls in computing using familiar concepts, such as puzzles, customization, and story-telling. Given the young age of our users, traditional means of computer science instruction, such as text-based directions and a focus on a specific programming language cannot be used. Children learn best using a visual programming environment such as Blockly that allows them to complete tasks, communicated via text and picture-based instructions, using familiar concepts like puzzles. Children are also more engaged in games that appeal to their unique interests and tell a story, facets that can be integrated into an educational game to motivate and sustain user interest in the concepts being taught.

Although user testing shows that *Code Girl* engages and interests young girls in computer science, results also indicate that the game may be too complex for some users, such as those who are at a lower reading level or those who unfamiliar with the concept of a visual programming environment, to complete without assistance. As such, *Code Girl* may be most effective at engaging and inspiring slightly older users that initially targeted, or changes need to be made to better accommodate users who are less familiar with the vocabulary used or gaming environment provided.

Additionally, users may not be fully aware of the skills and concepts being taught in the game's challenges, but when prompted, demonstrate a knowledge and understanding of the computer science and logic lessons included in these challenges. The challenges are thus not as obviously educational as originally thought, but still convey the simplified and abstracted concepts they were designed to teach.

Users, furthermore, have more diverse interests than were originally accounted for. Many users demonstrated more of an interest in the customization portion of the game, but some indicated that they liked the

problem-solving of the challenges and wanted more challenges covering topics like math or more advanced programming. Additionally, participants continually suggested ideas for clothes and accessories like a crown or a football, that indicated their unique passions that can be included to make *Code Girl* appeal to an even broader audience.

With regards to technology, over the course of developing *Code Girl*, it became clear that certain optimizations to the application were necessary. When system testing was first performed, some avatar customization levels took close to a minute to load because of the amount of images that needed to be retrieved and the length of the JavaScript files, especially the file for rendering the clothes and accessories. This problem was solved by the use of lower resolution graphics and compressed JavaScript files to decrease the time it took to load the application when hosted on Google Apps Engine. Repeated testing of the application, both by the developers and by potential users, was critical for catching and addressing these errors and performance limitations.

6.3 Project Assessment

An assessment of the disadvantages and advantages of *Code Girl*, as it exists today, in introducing girls to computer science is provided.

6.3.1 Disadvantages

With the target age range of five to eight year old girls we chose for our application, there is a substantial difference in reading comprehension and logic skills among users. As such, the application, in particular the challenges, can be difficult for some users, yet very easy for others. It seemed in user testing, however, that prior experience with a visual programming environment was a better indicator than age of how quickly users progressed through the application. Additionally, young children are more familiar with playing games on mobile devices than they are with playing games on a desktop or laptop, thus a native mobile application may be a more effective means of engaging them in computer science than a web-based application, such as ours. To address this weakness, we could make a native mobile version of *Code Girl*, in addition to the responsive web-based application we have already created. Finally, not all girls who play *Code Girl* may be engaged in the game and the concepts it teaches, even if they enjoy customizing an avatar, because people have a variety of interests, thus *Code Girl* may not successfully interest all users in computer science and inspire them to pursue technology.

6.3.2 Advantages

Our application nevertheless appears to be effective at interesting young girls in computer science. As we saw in user testing, the girls were engaged in the game and excited to learn that by completing challenges to customize their avatar, they were learning computer science concepts and skills. Participants reported that they felt they learned something while playing the game, referencing concepts such as patterns, repetition, and loops. Moreover, many of the girls reported that they enjoyed playing the game, and two were actually disappointed that they finished the game quickly, indicating that the superhero story and customization incentive successfully engage our target audience in learning basic computing concepts. *Code Girl* may not interest all users, but it does appeal to most users with its use of engaging cartoon-like characters and story-telling, suggesting that it would be successful at a larger scale.

6.3.3 Objectives Met

Additional research should be conducted to further assess user satisfaction with *Code Girl* with respect to factors such as age and prior experience with programming games, but initial testing suggests that *Code Girl* successfully engages our target audience with appealing characters, educational problems that will likely challenge users, and customization. In our application, successfully completing challenges unlocks new accessories, incentivizing game play, so our users continue learning new concepts while dressing up their avatar as they would a doll, appealing to their unique interests while hopefully educating, empowering and inspiring them.

Appendix A

Recruitment Flyer



CODEGIRL

Code Girl is an educational, computer science game designed specifically for girls.

Data
Learning Commons, Room 203, Santa Clara University
Time

OUR STORY

Code Girl was developed by three Santa Clara University students during their senior year for their Senior Design project. It is now being further developed and researched by two of the original team members, Amanda Holl and Paige Rogalski, as part of a Master's Thesis.

OUR MOTIVATION

Today, fewer than 1% of women in college are majoring in computer science. We, as female programmers feel that this needs to change. We have designed our application to hopefully inspire more girls to learn how to code and pursue careers in computer science and technology.

HOW YOU CAN HELP

We would like to introduce our application to potential users to get feedback on improvements we can make to help our game better achieve its goals and to get girls interested in learning about programming. To do so, we will be conducting a usability evaluation followed by a focus group at the above data, location, and time. If you are interested in having your child participate in this study, please fill out the attached parental consent form and have your child bring the form with her to study. Please feel free to contact us should you have any questions.

Amanda Holl: 650.465.0421, aholl@scu.edu

Paige Rogalski: progalski@scu.edu

Appendix B

Usability Evaluation Introduction

My name is [Researcher Name] and this is [Researcher Name], and we are working on our Masters degrees in Computer Science and Engineering.

Thank you for coming in today! You will spend the next hour playing an online game called Code Girl. Your goal is to play until you have reached level three, but feel free to keep playing after you have reached level three. Do not worry if you do not reach this level. Please let us know if you have any questions as you play the game.

If you do not mind, I would like to make a recording of your screen as you play the game. [Child Assent Form]

After you are done playing the game, you will answer some questions about the game. Please let us know if you do not understand any of the questions or if you have any other questions.

Also know that you are free to leave the study at any time, or take a break if you need to.

Thank you. You may begin playing the game.

Appendix C

Usability Evaluation Survey

C.1 Basic information about yourself

Question 1: What is your age?

Open ended answer

C.2 Your experience with programming games

Question 2: Have you ever programmed or coded before?

Yes/No Answer

- Yes
- No

Question 3: Have you played any of these games?

Multiple Choice Question

- Scratch
- Alice
- Code.org
- Other

Question 4: Do other people in your house play these games or do computer programming?

Yes/No Answer

- Yes
- No

Question 5: Were you interested in programming before playing the game?

Yes/No Answer

- Yes
- No

C.3 Code Girl usability evaluation

For Questions 6-10, please circle your agreement with the following:

Question 6: I liked the clothes and accessories.

Visual Scale



Question 7: The challenges were hard.

Visual Scale



Question 8: I liked dressing up Grace.

Visual Scale



Question 9: I am interested in programming after playing the game.

Visual Scale



Question 10: I would play Code Girl again.

Visual Scale



Question 11: Any other comments?

Open-ended comment box

Appendix D

Focus Group Outline

My name is [Researcher Name], and I am working on my Masters degree in Computer Science and Engineering.

Thank you for coming in today! We will spend the next hour talking about your experiences with using Code Girl. I understand that you all have just spent the past hour dressing up Grace and playing challenges, and we would like to hear what you think of the game.

If you do not mind, I would like to make an audio recording of our focus group. This will allow me to go back a later time and review your comments so I do not miss anything.

Your honesty is greatly appreciated, and if you do not have an opinion or answer to any of the questions posed, please feel free to say so.

Also know that you are free to leave the group at any time. Please also stop me if you have any questions. Now, please introduce yourselves, telling us your name and what your favorite game is.

D.1 Research Questions

1. What did you like most about Code Girl? What did you like least about Code Girl? *[Objective: Assess if users in general like our product and identify specifically what they do or not like about it]*
2. What problems or challenges did you run in to? *[Objective: Identify issues preventing users from enjoying or benefiting from the application]*
3. What would you add to Code Girl? What would you remove from Code Girl? *[Objective: Learn how we can better meet users wants or needs]*
4. Did you learn anything playing the game? If so, what did you learn? *[Objective: Assess if we are meeting our goal of educating and inspiring users]*
5. Any other comments?

Appendix E

Complete Usability Survey Results

Note the use of the following short-hand notation: "SA" for "Strongly Agree," "A" for "Agree," "N" for "Neutral," "D" for "Disagree," and "SD" for "Strongly Disagree."

	Session 1			Session 2		
	1	2	3	1	2	3
What is your age?	7	7	6	9	8	7
Have you ever programmed or coded before?	Yes	Yes	Yes	Yes	Yes	Yes
Have you played any of these games? [Scratch, Alice, Code.org, Other]		Scratch	Alice	Scratch, Code.org	Code.org, other	Scratch
Do other people in your house play these games or do computer programming?	No	Yes	No	Yes	Yes	Yes
Were you interested in programming before playing the game?	Yes	Yes	Yes	Yes	Yes	Yes
I liked the clothes and accessories.	SA	SA	SA	SA	SA	SA
The challenges were hard.	SD	N	N	A	SD	A
I liked dressing up Grace.	SA	SA	A	SA	SA	SA
I am interested in programming after playing the game.	D	SA	SA	SA	SA	SA
I would play Code Girl again.	I don't know. (N)	SA	A	SA	SA	SA
Any other comments?	I don't know any comments			I think it was a great game. I liked all the things.		

Bibliography

- [1] C. Corbett, C. Hill, A. St. Rose. "Why So Few? Women in Science, Technology, Engineering, and Mathematics", American Association of University Women, Washington, DC, 2010.
- [2] Girls Who Code – Join 40,000 Girls Who Code today!, *girlswhocode*. [Online]. Available: <https://girlswhocode.com/>.
- [3] Scratch - Imagine, Program, Share, *Scratch*. [Online]. Available: <https://scratch.mit.edu/>.
- [4] The Alice Project?, *Alice.org*. [Online]. Available: <http://www.alice.org/index.php>.
- [5] Made with Code — Google, *Made w/ Code*. [Online]. Available: <https://www.madewithcode.com/>.
- [6] Coding for Kids, *Tynker.com*. [Online]. Available: <https://www.tynker.com/>.
- [7] ScratchJr - Home, *ScratchJr*. [Online]. Available: <https://www.scratchjr.org/>.
- [8] Learn on Code Studio, *Code.org*. [Online]. Available: <https://studio.code.org/>.
- [9] Best Educational Toys: Kids Tablets, Handheld Games, Learning Systems — LeapFrog, *LeapFrog*. [Online]. Available: <http://www.leapfrog.com/en-us/app-center/>.
- [10] Polly Pocket Games, *Polly Pocket*. [Online]. Available: <http://www.pollypocket.com/en-us/games>.
- [11] *105-2 Hearing: S. 2326, Children's Online Privacy Protection Act of 1998, Serial No. 105-1069*. 1998.
- [12] "Blockly." Google Developers. [Online]. Available: <https://developers.google.com/blockly/>.
- [13] Google Developers, *Blockly Games*, GitHub. [Online]. Available: <https://github.com/google/blockly-games/wiki>.
- [14] "jQuery." JQuery. [Online]. Available: <https://jquery.com/>.
- [15] HTML5 Introduction, *HTML5 Introduction*. [Online]. Available: http://www.w3schools.com/html/html5_intro.asp.
- [16] "Google App Engine Documentation." *Google Developers*. [Online]. Available: <https://cloud.google.com/appengine/docs/>
- [17] "Cloud Storage." *Google Developers*. [Online]. Available: <https://cloud.google.com/storage/docs/>.
- [18] "Datastore - NoSQL Schemaless Database." *Google Developers*. [Online] Available: <https://cloud.google.com/datastore/>.
- [19] "Build Software Better, Together." *GitHub*.
- [20] D. Gotterbarn, S. Rogerson. "Computer Society and ACM Approve Software Engineering Code of Ethics", *Computer* 32.10, 1999, 84-88.
- [21] T. Bell, B. Gibson. "Evaluation of Games for Teaching Computer Science", Proceedings of the 8th Workshop in Primary and Secondary Computing Education. Ed.: ACM. Aarhus, Denmark, 2013.

- [22] F. F. Tsui and O. Karam, *Essentials of software engineering*, 3rd ed. Sudbury, MA: Jones and Bartlett Publishers, 2007.
- [23] "Inspiring Women in Computing". Vol.52: ACM. 2009.
- [24] J. Jenson, S. de Castell, S. Fisher. "Girls Playing Games: Rethinking Stereotypes", Proceedings of the 2007 conference on Future Play. Ed.: ACM. 2007.
- [25] G. Gallavin, S. de Castell, S. Fisher. "UX For Kids' Products: Designing For The Youngest Of Users", UserTesting Blog. 2015.
- [26] "Usability Testing With Kids And Teens", Usability.gov. 2016.
- [27] Age requirements on Google Accounts, *Accounts Help*. [Online]. Available: <https://support.google.com/accounts/answer/1350409?hl=en>.
- [28] N. von Herten, *html2canvas*, *html2canvas*. [Online]. Available: <https://html2canvas.hertzen.com/>.
- [29] *HTMLCanvasElement.toDataURL()*, *Mozilla Developer Network*. [Online]. Available: <https://developer.mozilla.org/en-us/docs/web/api/htmlcanvaselement/todataurl>.
- [30] D. Davis, *download.js*, *download.js*. [Online]. Available: <http://danml.com/download.html>.
- [31] S. Wanderski, The Responsive jQuery Content Slider, *bxSlider*. [Online]. Available: <http://bxslider.com/>.
- [32] ISO-9241 Efficiency metrics, *UI Designer*. [Online]. Available: <http://ui-designer.net/usability/efficiency.htm>.
- [33] A. Holl, *Code Girl*, Santa Clara University Protocol, Santa Clara, CA, Protocol 16-03-765, Apr. 2016.
- [34] System Usability Scale (SUS), *Usability.gov*. [Online]. Available: <https://www.usability.gov/how-to-and-tools/methods/system-usability-scale.html>.