Marvin's Marine Adventures

Deanna L. Scinta

Follow this and additional works at: https://scholarworks.rit.edu/theses

Recommended Citation

This Thesis is brought to you for free and open access by RIT Scholar Works. It has been accepted for inclusion in Theses by an authorized administrator of RIT Scholar Works. For more information, please contact ritscholarworks@rit.edu.
Rochester Institute of Technology

"Marvin's Marine Adventures"

by

Deanna L. Scinta

Date: October 20, 1995
This volume is the property of the Institute, but the literary rights of the author must be respected. Please refer to permission statement in this volume for denial or permission, by author, to reproduce. In addition, if the reader obtains any assistance from this volume, he must give proper credit in his own work.

This thesis has been used by the following persons, whose signatures attest to their acceptance of the above restrictions.

<table>
<thead>
<tr>
<th>Name</th>
<th>Address</th>
<th>Date</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Rev. 10/93
Thesis Approvals

Thesis Committee:
Chief Advisor: Professor James Ver Hague/
Date: 11-16-95

Associate Advisor: Professor Gordon Goodman/
Date: 11/19/95

Associate Advisor: Professor Robert Keough/
Date: 10-20-95

Associate Advisor: Dr. Jeffrey Porter/
Date: 11-20-95

Department Chairperson:
Date: ______________________

I, hereby grant permission to the Wallace Memorial Library
of RIT to reproduce my thesis in whole or in part. Any reproduction will not be for
commercial use or profit.

OR

I, ________________________, prefer to be contacted each time a request for production is made.
I can be reached at the following address:

________________________________________
________________________________________
________________________________________
________________________________________

Date: ______________________
Introduction

"Marvin's Marine Adventures" is a prototype of an interactive game for 11-12 year old children, which aims to entertain and educates them on the marine world. In the information section of the game, facts are presented about eight marine vertebrate animals along with the habitat, distinguishing characteristics and ocean depth at which these animals can be found. This knowledge is then applied to the interactive game.

The objective of the interactive game is to lead Marvin on marine adventures as he dives to different ocean depths in search of the answers to intriguing questions which describe the unknown animal. In the process, children will engage in hypothesis testing, utilize cross referencing skills and apply information they’ve learned to another.

The development and function of the elements of "Marvin's Marine Adventures" can be discovered by discussing the evolution of my ideas in creating this interactive prototype. I will reveal construction of the elements, development of the main character, Marvin, and technical issues that were encountered. The application of the literature used in this prototype will also be discussed. Lastly, I will conclude with the results of testing the project and plans for the future of "Marvin's Marine Adventures."
Introduction

"Marvin’s Marine Adventures" is a prototype of an interactive game for 11–12 year old children, which aims to entertain and educates them on the marine world. In the information section of the game, facts are presented about eight marine vertebrate animals along with the habitat, distinguishing characteristics and ocean depth at which these animals can be found. This knowledge is then applied to the interactive game.

The objective of the interactive game is to lead Marvin on marine adventures as he dives to different ocean depths in search of the answers to rhyming questions which describe the unknown animal. In addition to learning about vertebrates, children will engage in hypothesis testing, utilize cross referencing skills and apply information from one context to another.

The development and function of the elements of “Marvin’s Marine Adventures” will be covered by discussing the evolution of my ideas in creating this interactive prototype. I will reveal construction of the elements, development of the main character, Marvin, and technical issues that were encountered. The application of the literature used in this prototype will also be discussed. Lastly, I will conclude with the results of testing the project and plans for the future of “Marvin’s Marine Adventures.”
Development and Function of the Elements

“Marvin’s Marine Adventures” is an interactive prototype that provides information on characteristics of eight marine animals and their habitat. It also supports a method to evaluate the learning of material presented. This prototype is divided into two sections: the information section and the interactive game. The information section provides the knowledge base for learning about these marine animals, while the interactive game evaluates the material that users explore.

Chapter 1

Development and Function of the Elements

Information Section

The information section of “Marvin’s Marine Adventures” is the area of the prototype where users explore factual material. It provides a knowledge base to answer questions in the interactive game element. This ensures that users are able to play the game without prior knowledge of the eight vertebrate marine animals chosen. This section not only serves as a knowledge base before playing the game, but acts as a cross reference resource during gameplay.

This section provides marine vertebrate material at three different levels of difficulty/specificity. Each of these levels are color coded in three ways: a level indicator that is located on the upper left hand corner of the screen, the cursor and the letter “M” in the submarine of the “menu” button. All three of these elements change color as users navigate from level to level. Yellow is the most general level, containing general vertebrate information. The orange level signifies the mid-level that provides information on marine vertebrate groups: mammals, marine reptiles, bony fish and cartilaginous fish. The greenest level or red level encompasses specific animal characteristics and habitat information of the eight marine animals.

Reference [1]:

Another general level reviewed basic vertebrate information. [Review Note: This level once contained invertebrate information, but this feature was removed due to the large amount of material this level would have contained. At this level, general vertebrate characteristics are reviewed. Also, there are “go to next” options at the bottom of the viewing periscope to enter the game.]
Development and Function of the Elements

“Marvin’s Marine Adventures” is an interactive prototype that provides information on characteristics of eight marine animals and their habitat. It also supports a method to evaluate the learning of material presented. This prototype is divided into two sections, the information section and the interactive game. The information section provides the knowledge base for learning about these marine animals, while the interactive game evaluates the material that users explore.

“Marvin’s Marine Adventures” may prove to be a valuable educational tool through the development of the elements in this prototype. This chapter reveals the necessary elements used and their function for the success of “Marvin’s Marine Adventures.” (Please refer to Appendix A for the complete Lingo scripts)

Information Section

The information section of “Marvin’s Marine Adventures” is the area of the prototype where users explore of factual material. It provides a knowledge base to answer questions in the interactive game element. This ensures that users are able to play the game without prior knowledge of the eight vertebrate marine animals chosen. This section not only serves as a knowledge base before playing the game, but acts as a cross reference resource during game play.

This section provides marine vertebrate material at three different levels of generality/specificity. Each of these levels are color coded in three ways: a level indicator that is located on the upper left hand corner of the screen, the cursor and the letter “M” on the submarine of the “menu” button. All three of these elements change color as users navigate from level to level. Yellow is the most general level, containing general vertebrate information. The orange level signifies the mid-level that provides information on the four vertebrate groups: mammals, marine reptiles, bony fish and cartilaginous fish. The lowest level or red level encompasses specific animal characteristics and habitat information of the eight marine animals.

Yellow Level:
The most general level reviewed basic vertebrate information. (Please Note: This level would also contain invertebrate information, but this feature was beyond the parameters of this prototype.) At this level general vertebrate characteristics are revealed. Also, there are “word” or “text” options at the bottom of the viewing periscope to enter the orange
level. These options represent a link to more specific information of the vertebrate groups. Users click on these words to explore the area selected in more detail. The vertebrate areas can be accessed by clicking on the "Map" button, which is located directly below the level indicator or clicking on the "Sub" button to view the previous level of material. (Please refer to figure 1)

Figure 1: An information screen of "Marvin's Marine Adventures."

**Orange Level:**
This middle level discloses material about selected marine animals groups; mammals, marine reptiles, bony fish and cartilaginous fish. To navigate through this level users click on the left and right arrow buttons, which are located at the bottom of each side of the periscope. For example, clicking on the word "mammals" displays photographic images, mammalian characteristics, and habitat trends. At the end of a level, users can navigate to the most specific level or red level, or view a different section by accessing the "map" or "menu" buttons.
Red Level:

To view this most specific level, users click the “animal” icons located at the bottom of the periscope in the orange level. These icons are small representations of the animals that are used in the interactive game. For instance, in the mammalian section, users obtain more information about orca whales or dolphins by clicking on their respective icon. If orca whales are selected, animal characteristics and habitat of this type of whale are presented. In addition, users have the opportunity to view a QuickTime movie of each animal, by clicking on the “movie” button. This five–ten second video clip allows users to view the marine animal in its environment.

Function:

The Lingo programming for this section is straightforward. To navigate from level to level the “go” command is used. Buttons or words that are clicked send users to specific locations by accessing labeled “markers” on the score palette in Macromedia Director. For example, if users click on the word “Cartilaginous Fish,” the script go “cartilaginous” sends the playback head of the score to the cartilaginous fish section of the prototype.

Global variables are commonly used to return users to a previous viewed screen. For example, the map button is clicked and the gMap variable is referenced. This variable stores the frame number into the container, gMap. To return to a previously viewed section, gMap is accessed with the script go to frame gMap. Similar Lingo functionality is used in the “Menu,” “Map,” “Help,” “Quit” buttons and “Words” within the periscope.

The red level of the information section includes a QuickTime movie about each marine animal. To view this movie, the “movie” button is clicked. The script for this button utilizes the “go” command and sends the playback head of the score to the next frame. An animation of Marvin pulling down a movie screen occurs and the movie is played. To ensure that the animal icons at the bottom of the screen are not visible while the movie is playing, the puppetsprites are set to false. Prior to returning to a specific marine animal screen, the puppetsprites are set to true. This enables users to view the animal icons at the red level.

Interactive Game

To view the interactive game for the first time, the global variable gMMA is set equal to “Load MMA.” This will send the playback head to marker “Load MMA” of the score in the Macromedia Director movie “MMA.” This allows users to view the introduction of the interactive game. To refer to different sections of “Marvin’s Marine Adventures” and then return to the game, a different Lingo script is accessed. For example, users can referenced material in the information section and return to the interactive game. The frame
The game begins with a question asking if instructions are needed to play the game. If the response is “yes,” users view the help section of the game. If the response is “No,” then a random “Who Am I?” question will be displayed on the computer screen. These questions are based on the material presented in the information section of “Marvin’s Marine Adventures. They focus on the animal characteristics and habitat information presented in the “Information” section. The question appears as a voice-over reads it. The text also highlights in “red” to help users follow along with the voice. This assists users in reading the question, especially if they are having difficulty pronouncing unknown words. At this point, users can can do one of the following: repeat the question, select an ocean depth, obtain an additional hint, access the dictionary, cross reference material in the information section, view the help section or exit the application. Note: The functionality of the interactive game is included with navigational elements. (Refer to figure 2 to view interactive game screen.)

Figure 2: A sample screen of the interactive game in “Marvin’s Marine Adventures.”
"Replay" Button

The "Replay" button allows users to repeat questions. Again, a voice-over is generated along with the animation sequence of the highlighted text. This button was implemented to help users pronounce and process difficult words, and to assist them in reading questions. Users have the opportunity to repeat questions as many times as necessary, which allows for the freedom processing of information at different rates.

Function:
The frame number of the score is stored in the global variable gReplayQ. This number is obtained at the beginning of the animation sequence of the question. When the "Replay" button is clicked the gReplayQ variable refers to the frame that initiates the animation sequence. The question is repeated with highlighted text and voice-over reading of the question. (Please refer to Appendix A for the complete Lingo script)

"Ocean Depth" Slider

The "Ocean Depth" slider bar is a navigation tool that consists of a slider or submarine and three color coded waves that represent different ocean depths. The different ocean depths are the surface, middle and bottom depths. For example, the highest wave or surface wave is the lightest shade of blue. In contrast, the lowest wave or bottom wave is the darkest shade blue. To choose an ocean depth, users must click and drag the submarine to the respective wave. The ocean depth is selected by lifting up on the mouse. The periscope closes and re-opens at the chosen depth. Users also have the opportunity to change ocean depths at any point during game play.

This slider bar is one of the most important features of the game because users must select an ocean depth in order to view marine animals in their specific habitats. Marine animals will not appear in ocean depths that they do not live. Based on their reading of the "Who Am I?" questions, users need to make an educated decision on which ocean depth to choose by utilizing the "hint" button or cross-referencing material in the information section of "Marvin's Marine Adventures."

There are three different ocean depths that users view; surface depths, mid-depths and bottom depths. The surface depths contain animals that inhabit waters from sea level to 3,000 feet. The vertebrate animals chosen for this ocean depth are dolphins, tuna, manta ray and hammerhead sharks. Orca whales, sea turtles and tuna can be found at
mid-ocean depths, which range from 3,000 feet to 5,000 feet. Tuna are found at both surface and middle ocean depths. Due to their habitat patterns; diving into deeper waters for food, but residing in surface waters. To avoid confusion, they are included in both ocean depths. The bottom-ocean depth ranges from 5,000 feet to the ocean floor and includes marine animals such as sea snakes and flounder.

The “Ocean Depth” slider bar was added to create the metaphor of actually travelling to different ocean levels. Due to programming complications, I was unable to have Marvin travel to ocean depths by clicking and dragging him around ocean waters. (Please refer to pages 32–36 for complete description of technical issues).

In addition to the ability to travel to different ocean depths, users are able to review “Who Am I?” questions by clicking and dragging the sub above the surface depth wave at any point during the game.

**Function:**
The submarine of the “Ocean Depth” slider bar moves vertically within a designated location on the screen. This range is determined by the “X” and “Y” coordinates. The points 0,0 refer to the upper left corner of the screen as opposed to the points 640, 480 in the lower right corner of the screen. These points are determined by typing “put the locV of sprite 44” into the message box of Macromedia Director. (Please Note: Sprite 44 is the channel in the score where the submarine is placed.) The vertical location is displayed in the message box after the return key is selected. The sub is designed to move vertically from point 24 to 83. To prevent it from travelling beyond this location, a constraint is established by programming the vertical location of the sub to equal these points.

The global variable gDepth is set to equal the vertical location of the submarine or sprite 44. When gDepth is within a certain range of points the words “Surface,” “Middle,” or “Bottom” will appear as the submarine vertically moves along the respective waves. This is accomplished by swapping cast members. On the mouseUp, the value of gDepth determines which ocean depth is to be viewed. For example, if the vertical location of the submarine is between points 28 and 46, the surface depth has been chosen. Users view an animation of the periscope closing and re-opening showing Marvin and the marine animals that inhabit surface waters. (Please refer to Appendix A for the complete Lingo script)
The "Hint" button was developed to provide users the opportunity to obtain an additional clue without returning to the information section of "Marvin's Marine Adventures." All of this material was previously presented in the information section. The first hint or "optional" hint helps users refer to the appropriate ocean depth that the unknown marine animal inhabits.

In addition to the "optional" hint, users may also encounter a "forced" hint. This hint automatically appears when an incorrect choice is made on the second attempt, after the answer is checked. This hint contains information about the unknown animal’s physical characteristics, as opposed to its habitat.

**Function:**
A question is selected at random when the unknown animal name is stored in the global variable gName. The "Hint" button refers to gName by storing this variable into gButtonHint. This adds the word "but" after the animal name to distinguish that it is a button hint and allows users to view the appropriate clue. *(Please refer to Appendix A for the complete Lingo script)*

**Marine Reptiles:**

*Sea Turtles*
Hint 1: The slow metabolic rate of these marine animals allows them to submerge in mid-ocean depths up to several hours.

Hint 2: These reptiles can not retract their heads or limbs into their shells.

*Sea Snakes*
Hint 1: This marine reptile is covered with smooth scales and lives along the ocean floor.

Hint 2: This marine reptile is equipped with a small head and deadly fangs.

**Bony Fish:**

*Tuna*
Hint 1: This fish inhabits surface waters, but is known to to dive deeper in search of food.
Flounder
Hint 1: This animal is lives along the ocean floor.
Hint 2: This bony fish has the ability to blend in with its environment.

Cartilaginous:

Hammerhead Sharks
Hint 1: These marine animals live in surface waters and travels in schools.
Hint 2: They make distinct swimming patterns, which is believed to be a form of communication.

Manta-Ray
Hint 1: These 3,000 pound animals like to swim in warm, surface waters.
Hint 2: When feeding, these marine animals unroll their head fins to funnel food to their mouth.

Mammals:

Dolphins
Hint 1: This marine mammal likes to live in warm, surface waters.
Hint 2: These warm-blooded animals are known to exhibit social behaviors toward human.

Killer or Orca Whale
Hint 1: These mammals are usually found in mid-ocean depths and travel in groups or "pods" of 20 or more as they hunt for food.
Hint 2: These mammals are known to be the most powerful and ferocious marine animals.
“Check Answer” Button

The “Check Answer” button is selected after users intersect Marvin with a marine animal. They must click and drag Marvin to navigate him through ocean waters, then touch a marine animal that they suspect as the correct choice. Users then must click the “Check Answer” button to determine if the answer is correct.

If the answer is right on the first attempt, users view an animal profile of the correct answer. In addition, they earn one fish icon. This icon appears in the upper right corner of the screen under the word “Score:.” To win the game and view the correct answer screen users must select five correct answers on the first attempt. If the answer is incorrect, on the first attempt, users will view a warning message that instructs them to try again and/or encourages them to click the “Hint” button for more information about the unknown animal. A correct answer on the second attempt allows users to view the animal profile screen. However, a fish icon will not be rewarded. If an incorrect choice occurs on the second attempt, a “forced” hint of the unknown marine animal will be displayed. A correct answer on the third attempt will again allow users to view an animal profile, without earning a fish icon. If an incorrect choice occurs on the third attempt the answer will be revealed by displaying the correct animal profile. (Please refer to page 72 for more information about the “Animal Profiles”)

This button was implemented to develop a checking system as users maneuver Marvin in the ocean waters. The game is programmed to remember the last animal that users intersect while the mouse is held down. Users then click the “Check Answer” button to determine if the answer was correct. Without this system users could easily intersect other animals as they searched for the correct choice, allowing errors to occur.

Function:
To determine a correct answer, the “Check Answer” button is selected. This button distinguishes correct and incorrect choices through the use of global variables. The animal name or answer is stored in the variable gName after a question is selected at random. On the mouseUp script of Marvin, the name of the animal last intersected is stored into gGuess. The variables gName and gGuess are set equal to each other to determine the correct answer. A correct answer on the first attempt adds a value of one to the variable gCountrightanswer. This variable maintains the number of correct answers on the first attempt and also displays a fish icon. If the answer is incorrect on the first attempt, an invisible cast member is exchanged with a warning message that encourages users to click the “Hint” button. In addition, a value of one is added to gCountwronganswer,
which maintains the number of incorrect choices. If gName equals gGuess on the second
and third attempts users will view the appropriate animal profile, but gCountrightanswer
will not increase because it is only referred to on the first attempt. However,
gCountwronganswer continues to count incorrect guesses. On the second incorrect
attempt, the variable gHint displays the appropriate “forced” Hint. The third incorrect
attempt sends users to the wrong answer screen by means of the variable gMissed. This
variable stores the correct answer and displays the appropriate “Animal Profile.” (Please
refer to Appendix A for the complete Lingo script)

Animal Profiles

The “Animal Profiles” were added to the game to reiterate factual information that was
presented in the information section, “Who Am I?” questions and hints of “Marvin’s
Marine Adventures.” These profiles are made up of a large magenta fish that is located
within the viewing periscope. This colorful fish contains photographic images of the
highlighted animal, its name, distinguishing characteristics and habitat. The “Animal
Profiles” are presented to users regardless of a correct answer because it reinforces the
material previously presented. This is the information desired to be learned. (Please refer
to Figure 3 to view an “Animal Profile”)

Figure 3: An “Animal Profile” in “Marvin’s Marine Adventures.”
**Function:**
The “Animal Profiles” are viewed when the correct answer is chosen or after users make three incorrect choices. When the variables gName and gGuess are equal or gCountwronganswer equals three an “Animal Profile” is displayed. The appropriate “Animal Profile” is retrieved from the cast window and it is “swapped” with a small square on the right or wrong answer screens. This is accomplished by putting the variable gName into gProfile. The letter “P” is added after gProfile to differentiate between the cast member name and the correct answer. (Please refer to Appendix A for the complete Lingo script)

**Score**
To earn a score users must answer questions correctly on the first attempt. This score is measured in terms of fish icons. To win the game, users earn five fish icons. These icons are small representations of the magenta fish used for the “Animal Profiles.” The same fish was chosen because it is associated with the positive behavior of making a correct choice with the animal profiles.

**Function:**
The global variable gCountrightanswer functions as a counter. A value of one is added to the number of correct answers when gName and gGuess are equal. This variable only changes when users answer correctly on the first attempt. A fish icon is used to keep score. A total of five fish icons are necessary to win the game. (Please refer to Appendix A for the complete Lingo script)

**Dictionary**
The “Dictionary” button is used to further explain the meaning and pronunciation of unknown words. At any point during “Marvin Marine Adventures” users are able to access unknown words by clicking on this button. A school of twenty-six fish is viewed. Each fish symbolizes a different letter of the alphabet. Users click on the fish that begins with the letter of the unknown word. An enlarged fish will then appear with a list of words that begin with the chosen letter. Users click on the unknown word for a voice-over to pronounce and define the word, along with viewing a text definition. For example, the unknown word is “swimbladder.” Users click on the “S” fish, which displays all of the words that begin with the letter “S.” They then click on the word “swimbladder” for a definition and voice-over. To exit the dictionary section, click on the dictionary.
button. This returns users to the location they viewed last. (Please note: This section of “Marvin’s Marine Adventures” is still under construction.)

**Function:**
The “Dictionary” button is under construction and does not function. When users click on this button, a small square is exchanged with a warning message by swapping cast members.

**Help**
The “Help” button provides several functions such as explaining the objective of the game, offers a brief description of navigation and includes instructions to play the interactive game. (Please refer to Figure 4 to view a help screen.)

---

**Figure 2: A sample screen of the help section in “Marvin’s Marine Adventures.”**
Function:
The "Help" button is accessed by the "go" command. The statement go "Help" sends users to the marker "Help" to view this section. To return to the previously viewed screen the global variable gHelp is used. gHelp is "called" when users exit the help section and return to game play. The Lingo script used is "go gHelp." (Please refer to Appendix A for the complete Lingo script)

Information Section—Navigation Descriptions:
Level Indicator: Color coded indicator that reveals the level of generality/specificity to which the animals belong.

Map: A chart that shows the relationship of marine animals. It also is a way to navigate in the information section.

Menu: Allows you to view color coded categories within specific color coded levels.

Information: This section of "Marvin's Marine Adventures" contains the necessary information to play interactive game.

Dictionary: A place to look-up terms in "Marvin's Marine Adventures."

Game: Interactive game that explores concepts that are presented in the information section.

Help: Provides an explanation of the objective of the game, description of navigation and game play instructions.

Quit: Allows you to exit the game.

Interactive Game Section—Navigation Descriptions:
Sea Depth: Click and drag sub over sea depth to view where marine animals live. Note: Marine animals can only be seen at their specific sea depth.

Check Answer: To determine if your answer is correct, click the "check answer" button after touching a marine animal.

Hint: If you need an additional clue, click the "Hint" button.

Please Note: The information, Dictionary, Game, Help and Quit buttons perform the same function in both the information section and interactive game.
**Object of Interactive Game:**

The object of this game is to lead Marvin on marine adventures as he dives to different ocean depths in search of answers to rhyming questions which describe the unknown animal.

**Rules of Interactive Game:**

1. Click on Marvin to go on a sea dive.
2. Read “Who Am I?” question.
3. To repeat question, click the “repeat” button.
4. If you need to view marine animal facts, click on the “information” button.
5. Select an ocean depth by clicking and dragging the sub to surface, middle and bottom waters. Note: Animals can only be seen at their specific sea depths.
6. To answer questions click and drag Marvin until he touches a marine animal. Next click the “check answer” button.
7. If your answer is correct on the first attempt, you will be rewarded with a fish. If your answer is incorrect, keep searching! Remember: You can get a hint or review material in the information section.
8. If your answer is correct on the second and third attempts you will not be rewarded with a fish icon.
9. To win the game, you must get five of the eight questions correct. on the first attempt.
10. If less than five of the eight questions are answered correctly, then the game is over.
11. Start game over or exit from “Marvin’s Marine Adventures.”

**Quit**

The “Quit” button allows users to exit from the application.

**Function:**

The Lingo language used for the “Quit” button is simple. The “quit” command is used to exit the application. (*Please refer to Appendix A for the complete Lingo script*)
Who Am I? Questions

The “Who Am I?” questions challenge users inference-making and memory recall skills of the factual vertebrate information, allows for hypothesis testing and the ability to cross reference material. These questions are designed as end-rhymes, which are two to four lines in length. Distinguishing animal characteristics and habitat information of the marine animals are used to describe the unknown animal. Game play begins through Lingo programming selecting a random question.

**Function:**
A “Who Am I?” question is selected from a text field that contains a list of animal names. The first line of a field is selected and deleted from the original list. This word is stored in the variable gName. This variable determines which question is viewed on the screen. Each time a question is used an animal name is deleted from the list. This occurs until there are no more words on the list or the game is won. *(Please refer to Appendix A for the complete Lingo script)*

**Marine Reptiles:**

*Sea Turtles*
These cold-blooded aquatic reptiles can swim in the ocean for miles.
In these waters they search for the perfect mate, but return to land to lay eggs and continue their fate.

*Sea Snakes*
This marine reptile is known for its venomous bite, paralyzing its prey so they will not put up a fight.
It is found in deep waters and land, and often resides in the ocean’s sand.

**Bony Fish:**

*Tuna*
Its body is shaped like a crescent moon, and this bony fish is known for its ability to really zoom!
Flounder
Camouflaged by the sand and sea bed,
this bony fish has two eyes on one side of its head.

Cartilaginous:

Hammerhead Shark
This elasmobranch is known for its unusual design,
its unique shape allows it to seek prey in no time.
This animal travels in a school,
and distinct swimming patterns are used as a communication tool.

Manta-Ray
Known as the “devil fish” as it was once said,
its horn-like appendages extend from its head.
Its body is equipped with headfins curled up tight,
which unroll as they prepare to fight.

Mammals:

Dolphins
Putting on a show for those who dare,
these mammals spin and twist throughout the air.
They are also known to be highly intelligent,
as they communicate messages sent.

Killer or Orca Whale
This mammal is known for its towering dorsal fin;
when in battle, this powerful creature is sure to win!
Chapter 2

Construction of the Elements

Creating the Marine Animals and Marvin

In developing "Marvin's Marine Adventures" my original idea was to create graphic elements with Strata StudioPro, a three-dimensional modeling program. However, as I became more involved in the project I felt it would be best to create these elements in Adobe Illustrator and Adobe PhotoShop. These programs were preferred because the marine animals and Marvin were organic organisms, which are difficult to create in Strata StudioPro. In contrast, Adobe Illustrator and Adobe PhotoShop allowed me to create more realistic animals with smooth curves. In addition, Adobe Illustrator's ability to create blends and Adobe PhotoShop's ability to shade and manipulate images also added to the overall look of the marine animals created.

The outline shapes of the marine animals and Marvin were created with the pen tool in Adobe Illustrator. I would click and drag the pen tool to connect points to create smooth, organic shapes of the marine animals. To form the desired shapes, the bezier curves were manipulated. To simplify the animal shapes, their body parts were divided into basic geometric forms. For more complicated shapes, Adobe Illustrator's "pathfinder filters" were used to join objects together by either adding or subtracting overlapping shapes.

The appropriate colors were selected to fill the shapes created with color blends. For example, the body of the hammerhead shark varies in shades of gray. This blend was created by combining a light gray with a dark gray and then a light gray. This three phase blend was formed by manipulating colors in Adobe Illustrator's gradient palette. This blend was applied by selecting the gradient tool and then clicking and dragging it at a desired angle and length. The angle and length were two variables that were easily controlled with the gradient tool. This process was repeated for each geometric shape of the animals. After several shapes were blended together, they were grouped to their respective body parts.

Some of the animal's body parts, such as the eyes did not need to be blended. These parts were filled in by adding solid colors. These shapes were then grouped together.

To create animation sequences of the marine animals and Marvin, the original illustration was copied and pasted into a new document. The appropriate body part was rotated. For example, a dolphin fin would be moved in several steps to create the animation. The
Construction of Elements

Creating the Marine Animals and Marvin

In developing “Marvin’s Marine Adventures” my original idea was to create graphic elements with Strata Studio Pro, a three-dimensional modeling program. However, as I became more involved in the project I felt it would be best to create these elements in Adobe Illustrator and Adobe Photoshop. These programs were preferred because the marine animals and Marvin were organic organisms, which are difficult to create in Strata Studio Pro. In contrast, Adobe Illustrator and Adobe Photoshop allowed me to create more realistic animals with smooth, flowing contours. In addition, Adobe Illustrator’s ability to create blends and Adobe Photoshop’s ability to shade and manipulate images also adds to the overall look of the marine animals created.

The outline shapes of the marine animals and Marvin were created with the pen tool in Adobe Illustrator. I would click and drag the pen tool to connect points to create geometric shapes of the marine animals. To form the desired shapes, the bezier curves were manipulated. To simplify the animal shapes, their body parts were divided into basic geometric forms. For more complicated shapes, Adobe Illustrator’s “pathfinder filters” were used to join objects together by either adding or subtracting overlapping shapes.

The appropriate colors were selected to fill the shapes created with color blends. For example, the body of the hammerhead shark varies in shades of gray. This blend was created by combining a light gray with a dark gray and then a light gray. This three phase blend was formed by manipulating colors in Adobe Illustrator’s gradient palette. This blend was applied by selecting the gradient tool and then clicking and dragging it at a desired angle and length. The angle and length were two variables that were easily controlled with the gradient tool. This process was repeated for each geometric shape of the animals. After several shapes were blended together, they were grouped to their respective body parts.

Some of the animal’s body parts, such as the eyes did not need to be blended. These parts were filled in by adding solid colors. These shapes were then grouped together.

To create animation sequences of the marine animals and Marvin, the original illustration was copied and pasted into a new document. The appropriate body part was rotated. For example, a dolphin fin would be moved in several steps to create the animation. The
number of sequences varied depending on the animal and body part animating. For instance, Marvin's swim cycle was seven steps, as opposed to the three steps necessary for the sea turtle to swim. The number of steps was determined by experimenting with the movement and deciding on “what looked good.”

The files were then imported into Adobe PhotoShop after the animation cycles were completed. They were saved as Illustrator 5 files, to convert into an Adobe PhotoShop document. To open illustrator files in Adobe PhotoShop select “place” under the Edit menu and the illustrator file appears.

To avoid the “jaggies” or white edge commonly viewed along the edge of an illustration, the animation sequences were anti-aliased to the appropriate color background colors. For instance, sea snakes were anti-aliased to the bottom ocean water, whereas hammerhead sharks were anti-aliased to the surface water depth. Marvin was anti-aliased to all three ocean depth backgrounds because he was visible at all sea levels.

To touch up the images, I added highlights and shadows to make them look as realistic as possible. The airbrush tool was used at a 7%-12% pressure to add highlights to the illustrations. The color chosen depended upon the animal being highlighted. To make shadows, the burn tool was used at pressures ranging from 7%-25%. Again, the pressure strengths depended on the animal being manipulated.

The magic wand tool was used to select the background. Next, inverse was selected under “Selection” of the menu bar to highlight the animals. The magic wand tool selects just enough background color to anti-alias images to that color. This allows for a crisp image to be viewed in Macromedia Director. The selection was copied and pasted into Macromedia Director, as one large cast member. To divide the illustration into individual steps of an animation sequence, each animal was cut and pasted as separate cast members.

Creating the Navigation Tools

The Navigational tools that were located vertically along the left side of the screen were created in the same manner as the marine animals and Marvin. However, there are three differences. First, the images were anti-aliased to the background of the periscope as opposed to the water. Second, as users move the cursor over the buttons the word describing that particular button appears. These words were attached to a second set of buttons that were switched through Lingo programming. The third and final difference
was creating the inverse of the button when users click them. Copies of the original buttons were being depressed. The brightness was decreased to -30, which created the effect that the buttons were depressing. Again, the buttons were switched through the use of Lingo. (Please refer to Appendix A for the complete Lingo script)

Creating the Quicktime movies

The Quicktime movies were created by capturing video from several different educational videos. These clips were played in a VCR that was attached to an AV computer. The tape was digitized by recording it in FusionRecorder. These video clips were saved with a compression of “None” and imported into Adobe Premiere. The sound was deleted because it was unnecessary for the project. My interest was users viewing animals in their environment. Also, this saved memory and allowed for smoother running Quicktime movies.

To edit the video clips, extraneous parts were deleted with the “cut” tool in Adobe Premiere. A fade-in transition was added in the introduction and a fade-out transition was attached at the end of the clip. The video was saved and made into a movie by selecting “Make” in the menu bar and then “Movie....” These movies were created in 256 colors at fifteen frames/second and were five to ten seconds in length. I also chose a compression of “None,” to assure a high video quality. There were eight movies created, each representing the different marine animals chosen for “Marvin’s Marine Adventures.”

Creating the Images and text

The photographic images for the information section and “Animal Profiles” were found in a variety of resource materials used to create the project. These images were scanned at 300 dpi at 50% of their original size. They were scaled to the desired size, depending on the graphic layout for each screen created and the shape of the image itself. To clean up images, Adobe photoshop’s “despeckle filter” and “unsharp mask filter” were used to maximize image quality.

A new Adobe Photosphop document was created that consisted of the first layer as a background for the information section. The second layer was made up of scanned images. These images were copied and pasted into a third layer for a drop shadow effect. To create a drop shadow, these images were deleted to black and “inverse” was selected under “Select” on the menu bar. The inverse was filled with black ink. The selection was inversed for a second time and images were feathered four pixels. A final inverse was made and then deleted. As a result, black images remained with feathered edges. These
images were adjusted two pixels to the right and two pixels down to create the drop shadow effect. A two column grid was the final layer added to the Adobe PhotoShop document. The ink of this layer was set to transparent, which helped to align images to the grid. This grid helped to add consistency to the screen designs throughout the project.

The text was added after the images were in place. It was originally typed directly into Macromedia Director, which helped to evaluate the placement of the text. This made it easy to revise, along with copying and pasting it into Adobe PhotoShop at a later time. A sans serif font was used because it is easy to read on the computer screen. The font size for the body text was 14 point, Helvetica bold and 22 point, Helvetica bold for the titles of each section.

Finally, the text and images were then in put into place, the layers were flattened and saved as a Pict file format. The files were imported into the appropriate Macromedia Director movies.

Please Note: This same procedure was used for the “Animal Profile” section of the project, except the drop shadows for the images and text was placed on the large magenta fish.

Creating the Periscope Background and Submarine

The periscope background and submarine were created with a white background that was 480 pixels x 640 pixels in size. The Adobe PhotoShop “noise filter” was added two times and a “motion blur filter” was applied at a 30 degree angle to create a textured pattern. Eight shades of blue, three shades of green and one shade of red were also added with Adobe PhotoShop’s “variations filter.” This resulted in a background texture with a metallic blue appearance.

To create the viewing periscope, Adobe PhotoShop’s “circular marquee” tool was used. The shift key was held down to constrain the circle within the confines of the background, leaving enough room at the bottom and sides for navigation. This selection was then deleted to make a white circle. The edge of the viewing periscope was created with a slightly larger circle of the background texture. It was copied and pasted onto the background with the white circle. This circle was shifted up one pixel and one pixel the left; creating a shadow for the viewing periscope.
The metallic background was divided into sheets of metal by adding thin lines to the texture. These lines also served as division for the navigation. They were created with Adobe Photoshop's "pencil" tool at a 20% pressure with black ink. To create the shadow effect, a second white line with a 20% pressure was drawn one pixel to the right of the black line. Two horizontal lines divide the background into thirds and one vertical line signified the beginning of the periscope and end of the navigation portion of the background.

Along with adding dividing lines, rivets were also created. They were approximately 1/4 inch away from each other and the line itself. These rivets were made by using Adobe Photoshop's "circular tool" and applying the "glass lenses brighter" filter. These rivets were then copied and pasted along both sides of the lines.

Highlights were added to the background by using Adobe Photoshop's "blur" and "airbrush" tools. These tools were set at pressures under 10%, with medium sized brushes. The light source was in the upper left corner, therefore I used a sweeping motion that travelled from the upper left corner to the lower right corner with both tools.

Please note: Marvin's submarine was created using the above method. However, I added highlights and shadows to create the effect that the sub was round, giving depth or a three-dimensional appearance.

Creating the Ocean Backgrounds/Sky

The background for the information section was created from a scan of an underwater scene at 150 dpi and saved in Pict file format. This image was then blurred twice. The color balance was adjusted by adding +100 green and +35 blue. It was then saved at 72 dpi and imported into Macromedia Director.

A different section of the original underwater image was scanned for the introduction and quit sections of "Marvin's Marine Adventures." This section of the scan included a close-up of a fish, which was distorted with an Adobe PhotoShop filter to create a wave-like pattern. This image was blurred twice and the color balance was adjusted by adding +100 green and +35 blue in Adobe PhotoShop.

To make the waves look like they were continually moving, the first wave was copied and flipped horizontally. Through the use of Lingo, I programmed the waves to switch cast members; giving them the appearance of continually moving.
Appendix A for the complete Lingo script

The water background in the interactive game and sky of the introduction and quit sections were all created by making a blend with shades of blue. These colors varied depending on the particular background. For example, a light shade of blue and a medium shade of blue were used to make the sky and surface water backgrounds, while darker shades of blue were used to create the bottom ocean depth.

To make the backgrounds, the desired colors were chosen and Adobe PhotoShop's gradient tool was selected. This tool was clicked and dragged from bottom to top of the canvas, creating a smooth blend with even distribution of colors. Finally, the images were saved in Pict file format and imported into Macromedia Director.

To create the ocean floor, an area was selected with the lasso tool for the sand. This selection was filled with a tan color. Hilights and shadows were added with Adobe PhotoShop's airbrush tool. The rocks and plants were created with the lasso tool and the air brush was used to fill in the appropriate colors. The color and air pressure varied, creating realistic images. Lastly, Adobe PhotoShop's burn tool was used to create shadows in the sand of the plants and rocks.

Creating the Sound

The introduction sounds were recorded from Ocean Surf, "Pacific Ocean Surf" CD. This type of music was chosen to create the effect of being at the ocean, hearing waves crash into each other. In Macromedia SoundEdit 16, a second sound channel was added to create the submarine sound effect. This sound was a machine engine from a sound effects CD. The length of the sound was determined by the number of frames/second. The two sounds were mixed together, saved in AIFF format and imported into Macromedia Director. This sound was placed in the second sound channel of the score palette. The first sound channel of the score palette was used for the voice-over which gives a brief description of "Marvin's Marine Adventures."

In addition to the introduction sounds, Lingo was used to create "puppetSounds" for the clicking noises of buttons in the information section and interactive game. These sounds were reserved for the first sound channel, while the longer sounds were accessed in the second sound channel or as "sound Playfiles."

Along with the two sound channels in the score palette of Macromedia Director, "sound Playfiles" were accessed for the interactive game. These sound files were stored within
the same folder as the Macromedia Director movies and were represented as sound channels three through seven. Sound playfiles were helpful especially when long sound loops were used for music such as Sally Roger’s “Songs for the Whole Earth.” This sound was used as background music when users read the “Who Am I?” questions. “Sound playfiles” were also used for the background ocean music at all three ocean depths. A benefit of “sound Playfiles” was their quick loading time, which allowed little time for delay in game play.

Lastly, the voice-over in the information section, was similar to the voice-over that read the “Who Am I?” questions. These sound files were placed in the second sound channel. Again, leaving the first channel free for clicking noises of the buttons and the sound playfiles for the background music.
Development of "Marvin"

In developing Marvin, I had envisioned a character that would lead users through "Marvin's Marine Adventures." He would serve as a positive role model and also entertain users as he leads them through the information section of "Marvin's Marine Adventures." In addition, Marvin takes on an more active role in the interactive game as he searches the ocean waters by clicking and dragging him to find unknown marine animals.

Marvin possesses both human and animal traits because frogs are amphibian and have the ability to survive on land and in water. I wanted users to be able to relate to an animal that they had seen or even touched on land, but could also survive on water. His skin should be a bright green color due to his frog-like features. His arms and legs resemble human arms and legs; three of his fingers and feet are web-like flippers. Flippers are used for his hands and feet because they aid him in swimming around the ocean waters. His mid-body region is round; in fact he is considered to be "plump." I thought this would make him a lovable character. Located on Marvin's oval shaped head are his large, oblong eyes. These eyes are large, so he can use his keen eyesight to detect fish while navigating in the ocean waters.

Marvin's marine outfit is a simplified scuba diver's outfit. He is equipped with a blue and white scuba suit, which properly insulated as he swims in the ocean water. His yellow mask is permanently attached to his head, which surrounds his large eyes and that aid him in swimming. Marvin's mouth piece is always located on the inside of his mouth and moves upward to allow for breathing underwater. In addition, he is also equipped with two air tanks that are located on his back and attached by straps that are fastened around his waist and shoulders. (Please refer to Figure 5 to view Marvin.)

Figure 5: An illustration of Marvin, the main character in "Marvin's Marine Adventures."
Development of “Marvin”

In developing Marvin, I had envisioned a character that would lead users through “Marvin’s Marine Adventures.” He would serve as a positive role model and also entertain users as he leads them through the information section of the “Marvin’s Marine Adventures.” In addition, Marvin takes on an active role in the interactive game as he searches the ocean waters by clicking and dragging him to find unknown marine animals.

Marvin possesses both human and frog-like characteristics because frogs are amphibian and have the ability to survive on both land and water. I wanted users to be able to relate to an animal that they had seen or even touched on land, but could also survive on water. His skin should be a forest green color due to his frog-like features. His arms and legs resemble human arms and legs, however his hands and feet are web like flippers. Flippers are used for his hands and feet because they aide him in swimming around the ocean waters. His mid-body region is round; in fact he is considered to be “plump.” I thought this would make him a lovable character. Located on Marvin’s oval shaped head are his large, oblong eyes. These eyes are large, so he can use his keen eyesight to locate fish while navigating in the ocean waters.

Marvin’s marine outfit is a simplified scuba diver’s outfit. He is equipped with a dark grey scuba suit, which properly insulated as he swims in the ocean water. His yellow mask is permanently attached to his head, which surrounds his large eyes and that aide him in seeing. Marvin’s mouth piece is always located on the inside of his mouth and extends upward to allow for breathing underwater. In addition, he is also equipped with two air tanks that are located on his back and attached by straps that are fastened around his waist and shoulders. (Please refer to Figure 5 to view Marvin.)

![Figure 5: An illustration of Marvin, the main character in “Marvin’s Marine Adventures.”](image-url)
In developing “Marvin’s Marine Adventures” there were several technical issues that needed to be evaluated for this interactive prototype to function. These issues were the navigation design, user orientation within the project, changing the color palette to function at 256 colors and animation sequences of the marine animals.

The first and foremost concern was designing the navigation for “Marvin’s Marine Adventures.” In creating this project, it was important for users to be able to navigate through the information section and interactive game. It was essential to effectively communicate the functionality of “Marvin’s Marine Adventures” to users.

The navigational buttons were originally located around an oval-shaped periscope. This designed offered as much surface area as possible for users to view the game, leaving the edges of the periscope for navigation. However, it was difficult to develop navigational icons that were easily understood around the periscope. As a result, a circular periscope was created on the right side of the viewing screen, leaving the left side for navigation. This interface design was stronger because it added balance to the screen layout and also offered organization and consistency to the navigation.

To organize the buttons, a global and local navigation system was devised. Global navigation were buttons used in the information section and interactive game. In contrast, local navigation consisted of buttons used in either the information section or the interactive game. To distinguish the difference between global and local navigation the metallic background was divided with lines, exemplifying a metallic sheet-like appearance. For example, the bottom left corner of the background consisted of global navigation. The “Quit” and “Help” buttons were always placed at this location. The middle area of the background refers to the specific sections of “Marvin’s Marine Adventures” available to users; the “Information,” “Game” and “Dictionary” buttons. These buttons also serve as global navigation because their functionality was necessary throughout “Marvin’s Marine Adventures.” The top of the metallic background was reserved for local navigation because these buttons were specific to the section of “Marvin’s Marine Adventures” being viewed. For example, in the information section the color coded “Level Indicator,” “Map” and “Menu” buttons were included as local navigation. In the interactive game, the “Ocean Depth” slider bar, “Check Answer” and “Hint” buttons were considered as local navigation. (Please refer to Chapter 1 for navigation functionality)
The global/local navigational system was beneficial for three reasons. First, using buttons specific for the different regions on the screen allows for more navigational space within each section of “Marvin’s Marine Adventures.” Second, this system developed consistency for users. There was awareness that the upper left section of the screen was used for local navigation and the bottom two thirds were used for global navigation. Lastly, it was not necessary to have all of the buttons functional throughout “Marvin’s Marine Adventures.” The local navigation was applicable to the specific section only, whereas global navigation functioned throughout the prototype.

In addition to re-designing the interface, it was necessary to ensure that users understood their orientation within “Marvin’s Marine Adventures.” To avoid confusion in the information section, the local navigation was color coded. The color coded “Level Indicator” displayed the different levels to be viewed. This navigational element helped orient users within the information section. The “Map” displayed color coded graphics which represented the user’s locality. The letter “M” on the “Menu” button changed colors depending on the specific menu level that was viewed.

The colors yellow, orange and red represented each of the levels of information. Yellow or the highest level was chosen to symbolize general vertebrate information. This color was viewed as a light, happy color that welcomed users to “Marvin’s Marine Adventures.” A combination of yellow and red was orange, therefore it was the middle level of information. At this level, material about the specific animal groups was presented. These vertebrate groups were mammals, marine reptiles, bony fish and cartilaginous fish. Red was the lowest or most specific level, representing a warning to users that this was the final level of information. Material about the eight specific marine animals was revealed at this level.

The local navigation of the interactive game was also color coded. The “Ocean Depth” slider bar was color coded in shades of blue. This was implemented to assist users in distinguishing the three different ocean depths. The lightest shade of blue represented surface depths, middle depths were medium shades of blue and the bottom depths were distinguished as the darkest shades of blue.

The “Ocean Depth” slider bar helped create the effect that users were travelling to different ocean depths. This was accomplished by selecting an ocean depth, the periscope would close and then open at the new ocean depth. The ocean backgrounds were color coded to the colors of the slider bar waves. The lightest shade of blue or top wave represented surface depth. In contrast, the darkest shade of blue or bottom wave was the bot-
tom ocean depth. The deeper one travels in the ocean, the darker the water appeared.

This slider bar was implemented due to programming complications of navigating Marvin in the ocean waters. Marvin had the potential to travel long distances through Lingo programming. He was a moveable sprite that users would click and drag, appearing to swim endlessly in ocean waters. The ocean background was programmed to move 10 pixels in the opposite direction that Marvin was travelling. This created the effect that Marvin was swimming forward, but in reality the background was moving backwards. This functionality worked as planned, however it was difficult to animate the marine animals while Marvin was moving in the ocean.

The marine animals were originally animated by using “In-betweens.” This was a problem when Marvin began swimming distances farther than the screen because the background was continually changing, but the animals always remained on the screen. The animation sequences were specific to the 640 pixels x 480 pixel area, whereas Marvin was capable of travelling longer distances.

Another method was to create film loops, in which the registration points of each animal sequence was altered to create animated movements. This was an unsuccessful attempt because altered registration points created a bounding box that was larger than the viewing screen. The interactive game was programmed to check answers by determining the intersection of Marvin and the last animal he touches before clicking the “Check Answer” button. The marine animals were always intersecting Marvin due to the large bounding boxes of the film loops. Therefore, the answers were skewed and this was an invalid method of animating the marine animals.

A final method tested was to script the marine animals to move 10 pixels in the opposite direction of the background. This test also failed because the marine animals would appear to “jump” into position, causing rigid animation sequences.

Due to programming complications and time constraints, the functionality of “Marvin’s Marine Adventures” changed. As a result, the background no longer animated. The marine animals animated in and out of the 640 pixels x 480 pixel viewing screen. Marvin was a moveable sprite that users would click and drag around the viewing periscope. He automatically returns to the center of the periscope when users lift up on the mouse, to ensure that he would not get “stuck” in an area that was not visible to users. Lastly, the slider bar was implemented as local navigation to allow users to view different ocean depths.
Another navigation issue to be resolved was the left and right arrows at the orange level in the information section. At this level, users viewed information about the four major vertebrate groups; mammals, sea reptiles, bony fish and cartilaginous fish. Each of these sections included two to three screens of information. It was necessary to develop a method to navigate within the four animal sections or orange level. The left and right arrow buttons were placed in several positions before deciding on a final location. At first, these arrows were going to be included in the local navigation in the upper left corner of the screen. However, this position was too far away from the periscope. It was confusing and difficult to associate the arrows with the material presented. They were then placed on the middle of the periscope frame. Again, the arrows were too far away from each other and the material presented. Inside of the periscope was a third location that the arrows were placed. Unfortunately these arrows were too close together and the information was cramped. Finally, the arrows were positioned outside of the bottom of the periscope. Graphically, this position did not disrupt the overall design and they were close enough to the material and each other. This was not confusing to the users and enabled them to navigate within the orange level.

To make sure users knew which buttons to click on was a final navigational concern. It was essential for users to select an ocean depth after viewing a question to locate unknown marine animal. To avoid confusion, directions were included at the bottom of the periscope and warning messages were displayed when incorrect buttons were clicked. These messages encouraged users to click on the appropriate buttons to continue game play.

Besides navigation issues, color palette issues needed to be resolved. All of the images were created in thousands of colors. However, the more colors used, the slower the program functioned. To maximize functioning speed, an adaptive palette was created. This palette allowed the “Marvin’s Marine Adventures” to run at 256 colors. This not only improved the running speed, but it decreased the memory of the program by one third of its original size.

To create an adaptive palette, an Adobe PhotoShop document that contained all of the colors in the project was created. This file was converted from the RGB color mode to indexed color and adaptive palette under “Mode” of the menu bar. This file was saved and imported it into Macromedia Director. To install the adaptive palette in Macromedia Director, a warning message appears when importing the Adobe PhotoShop file. All of the cast members in Macromedia Director were converted to 256 colors by selecting “Transform Bit-Map” under the menu bar.
Two adaptive palettes were originally created; one for the information section and one for the interactive game. Unfortunately, as users navigated from section to section the screen flashed as the palettes changed. As a result, one adaptive palette was used for both sections of “Marvin’s Marine Adventures.” This palette would cause minimal loss in image quality. However, the slight loss in quality was more beneficial than slower functioning of the program that was three times the memory size.

Animating the marine animals was a final technical issue that was encountered. “Marvin’s Marine Adventures” ran slower when animation sequences were created with film loops. To improve the running speed, the appropriate cast members were manually switched with the film loops. These loops were then deleted from “Marvin’s Marine Adventures” and the animation sequences ran at the desired speed.
Chapter 5

Review of the Literature

Educational Psychology Research

In this chapter I will discuss the reasoning behind the chosen age group and the methods used for developing children's cognitive, social and self-esteem. The role that the computer has on users' eye-hand coordination and gender related issues will also be discussed.

The target audience for "Marvin's Marine Adventures" is 11-12 year old users. This age level was selected because of their often expressed high energy levels, enthusiasm and curiosity. It has been noted that 'boys generally like arcade style games, while girls are more attracted to exploration and problem solving games.' (Gardner, pg. 174) To appeal to both genders, interests of both groups were integrated in the development of the application. For example, the interactive game promotes problem solving as users answer questions. Users also need to match the marine characters that are swimming at various ocean depths. This way, both genders interested in "Marvin's Marine Adventures" can be engaged.

It was my goal to exercise users' cognitive, social and self-esteem. Cognitively, users engaged with this application with different learning styles. They are actively involved in the learning process, develop problem solving skills, generate hypotheses and discover methods of organizing information. Socially, this application helps users develop the ability to take on multiple roles. Positive reinforcers are also used to sustain users' self-esteem.

Cognitive Development

Children in the 11-12 year old age group are continually developing their cognitive skills and exhibit a variety of cognitive styles of learning. "Cognitive styles are the tendencies and preferences to respond to a variety of intellectual tasks and problems in a particular fashion. Some responses are impulsive, while others are reflective." (Lefrancois, pg. 134) Impulsive children are concerned with a quick response. They answer questions with the first thought that comes to mind. These answers are more likely to be incorrect. Reflective children take the time to think before they speak, they seem to evaluate alternative answers and give the correct answer, rather than a quick response. The development of "Marvin's Marine Adventures" promotes learning in a reflective manner. Incorrect hypotheses are penalized, as a result users are encouraged to take their time when reviewing information. Information is also easy to reference by simply clicking on the "Menu" button, "Map" button or "Word" categories to view material. Users are not hindered by
In this chapter I will discuss the reasoning behind the chosen age group and the methods used for developing children’s cognitive, social and self-esteem. The role that the computer has on users’ eye-hand coordination and gender related issues will also be discussed.

The target audience for “Marvin’s Marine Adventures” is 11-12 year old users. This age level was selected because of their often expressed high energy levels, enthusiasm and curiosity. It has been noted that “boys generally like arcade style games, while girls are more attracted to exploration and problem solving games.” (Gardner, pg. 174) To appeal to both genders, interests of both audiences were used in the development of the application. For example, the interactive game promotes problem solving as users answer questions. Users also need to react to the animating characters that are swimming at various ocean depths. This was specifically designed to keep both genders interested in “Marvin’s Marine Adventures.”

It was my goal to exercise users’ cognitive, social and self-esteem. Cognitively, users engaged with this application with different learning styles. They are actively involved in the learning process, develop problem solving skills, generate hypotheses and discover methods of organizing information. Socially, this application helps users develop the ability to take on multiple roles. Positive reinforcers are also used to sustain users’ self-esteem.

Cognitive Development

Children in the 11-12 year old age group are continually developing their cognitive skills and exhibit a variety of cognitive styles of learning. “Cognitive styles are the tendencies and preferences to respond to a variety of intellectual tasks and problems in a particular fashion. Some responses are impulsive, while others are reflective.” (Lefrancois, pg. 118) Impulsive children are concerned with a quick response. They answer questions with the first thought that comes to mind. These answers are more likely to be incorrect. Reflective children take the time to think before they speak, they seem to evaluate alternative answers and give the correct answer, rather than a quick response. The development of “Marvin’s Marine Adventures,” promotes learning in a reflective manner. Incorrect responses are penalized, as a result users are encouraged to take their time when reviewing information. Information is also easy to reference by simply clicking on the “Menu” button, “Map” button or “Word” categories to view material. Users are not limited to
viewing material only one time, it is recommended to review information as many times as necessary. There is no time limit to achieve five of the eight correct responses.

In developing cognitive skills, users engage in active learning. “An active learner’s behavior emphasizes asserting, volunteering and seeking information.” (Borich, pg. 161) Active learning is promoted throughout “Marvin’s Marine Adventures” when users click on elements, observe animation sequences and video clips, listen to audio feedback and participate in the interactive game. In the “Information” section, users navigate from section to section learning about the marine world. Material is presented at three different levels, ranging from most general to most specific information. They click on different elements to read, listen and view material about the marine environment. Active learning also occurs in the interactive game as users click and drag Marvin around various ocean depths in search of unknown animals. Similarly, users are challenged to answer five of the eight questions correctly as they read, listen and view different marine animals in their environment.

Along with engaging in active learning, users’ problem solving skills are exercised when answering rhyming questions in the interactive game of “Marvin’s Marine Adventures.” These questions were developed to creatively disclose the desired information to be learned. Only some of the marine animal’s characteristics and habitat facts were revealed to stimulate user’s thinking skills. This was designed to encourage users to rely on their ability to recall the factual material and apply this information to the questions in the game. If users need additional assistance, they can access the “Hint” button and cross reference material in the “Information” section of “Marvin’s Marine Adventures.” This type of assistance is available throughout the interactive game to help minimize errors and enable users to reach their goal of learning about the marine world by responding to questions correctly.

Users also formulate hypotheses by applying concrete or factual material to the creative manner of participating in the interactive game. Hypothesis generation occurs as users recall material from the “Information” section, review questions and access the “Hint” button. A hypothesis is created as users strive to organize and adapt information in an effort to seek a balance in one’s conceptual world. “Organization is the tendency of all individuals to systemize or combine processes into coherent systems. Schemes are organized patterns of behavior or thought that children formulate as the interact with the environment. Schemes are both behavioral and cognitive. The tendency to adjust to the environment, or adaptation, occurs as one encounters new experiences. This can be achieved through the processes of assimilation and accommodation. “Assimilation is the
process of fitting a new experience into an existing scheme, while accommodation is the process of creating a new scheme or revising an old scheme to fit the new experience.” (Lefrancois, pg. 53) These processes occur continuously throughout the user's participation in “Marvin's Marine Adventures.” In hypothesis testing, users touch marine animals with Marvin and then select the “Check Answer” button. This method of evaluating skills helps to stimulate thinking and increase interest in the subject matter.

Piaget states, “People thrive to organize their knowledge in order to achieve the best possible adaptations to their environment.” (Lefrancois, pg 53) This process is called equilibration. In contrast, disequilibration promotes learning when schemes are inconsistent with past and present thoughts. In order for equilibration to occur, disequilibration must occur. Disequilibration occurs spontaneously through individual maturation and experience or can be stimulated by someone else. “Meaningful learning occurs when people create new ideas, knowledge, rules and hypothesize to explain things, from existing information.” (Lefrancois, pg 56) To solve problems, it is necessary to search one’s memory for information that can be used to come up with a solution. Using information can mean experimenting, questioning, reflecting, discovering, inventing and discussing. Individuals need to create knowledge to solve problems and eliminate disequilibrium. This process occurs when users formulate their own thoughts to answer questions. These ideas are derived from material previously viewed and result in users potentially developing learning schemes. These schemes are evaluated when users are actively involved in the interactive game.

**Social Development**

In addition to the children’s cognitive development, they are also developing socially. Children at this age are developing the ability to undergo third person point of view or multiple role plays. They are learning to understand how other people feel. To take on the perspective of Marvin, children would further develop their ability to take on multiple roles. I hoped that they would imagine themselves as Marvin travelling through the information section and in the ocean waters of the interactive game. To introduce users into the marine environment, they would act as Marvin, the informational guide, then take an active role navigating the ocean waters in search of marine animals in the interactive game. Similarly, users emulate the marine animals, discovering what it might be like to a marine animal in the ocean. They learn about marine animal characteristics, along with gaining an understanding of their habitat.
Self-Esteem

"Marvin's Marine Adventures" also helps children's self esteem, through providing them positive reinforcement for their behaviors. Users are positively reinforced when questions are answered correctly by earning a fish icon and viewing an animal profile. A brightly colored fish appears and a positive "ding" sound is played. These profiles also reiterate the desired information to be learned. The name of the marine animal, distinguishing characteristics and habitat information are presented on a large magenta fish. When questions are missed after three attempts, users are still able to view the animal profiles; however, they will not earn a fish icon. By showing animal profiles after incorrect attempts, users are able to review desired information which may help increase their success rate in future efforts.

Eye-hand Coordination

In addition to developing users' cognitive, social and self-esteem, the computer also serves as a learning resource to exercise eye-hand coordination skills. Participation in the interactive game allows users to develop fine motor skills as they click and drag the mouse to different areas on the computer screen. Users focus on making the mouse move to the desired locations. To assist children with their motor skills it was necessary to develop "icons" that are large and easy to click on. As a result, users within this age group have a greater likelihood in being able to play the game.
Review of Scientific Literature

In developing “Marvin’s Marine Adventures” it was necessary to determine learning objectives for this prototype. The focus would be on eight vertebrate marine animals. Users would learn about typical characteristics of the animal group and then characteristics and habitat of the specific animal. For example, to review material about the orca whale, users were presented with material covering general vertebrate information, information about mammals and specifically orca whale characteristics and habitat material.

It was my goal to develop a hierarchical system for users to organize information to relate this material to the larger scope of the marine world. To view a description of material for all vertebrate animals please refer to the “Scientific Research” section on pages 42–47 and also the “Who Am I?” Questions on page 15. This will offer a complete information of research material and the chosen characteristics and habitat information used in “Marvin’s Marine Adventures.”

1. Marine Reptiles

Scientific Resources:

The scientific resource material covers life in the ocean, ranging from marine animals that inhabit areas near the top of the ocean waters to bottom of the ocean floor. These animals all vary in shapes, colors and sizes. For “Marvin’s Marine Adventures,” marine animals were chosen from different ocean depths to be presented in this project. Resources such as The Life of the Ocean were used to acquire strong visual imagery that will be used as references to create the animated marine animals. In addition the book, Sharks and Other Creatures of The Deep served as valuable resources for presenting informative educational material for the appropriate age level.

Along with studying marine animals of the ocean, it is also necessary to establish an environment in which these organisms live. In the book, The Sea I begin to gain an understanding of the distinguishing environmental characteristics at different ocean levels. As a result, this directly impacted the types of marine animals that inhabit a particular region of the ocean.
Scientific Research

I. Vertebrates

A. “Life on earth began over 4,500 million years ago. Yet only recent years have humans begun to explore the depths of the oceans. Almost every yard of dry land has been mapped and explored, but below the waves lies a world that remains largely unknown, which is home to all kinds of extraordinary creatures.” (Steele, pg. 4)

B. In the marine world, vertebrates tend to be larger and more active than other ocean dwellers, along with dominating their habitats. “All possess skeletal systems including a backbone or spinal column, well developed organ system and well developed senses.” (Miller, pg. 14) This group includes fish, reptiles, mammals, birds and amphibians. These animals thrive in all oceanic environments.

1. Marine Reptiles

Aquatic reptiles evolved from their terrestrial based cousins. They have lost many adaptations of the land dwellers, but have developed other features that enabled them to survive in water.

All reptiles are cold-blooded, most of their body heat must come from outside of their bodies. Water is capable of conducting heat far more quickly and efficiently than air. Lastly, marine reptiles require heat of tropical water or temperate seas.

a. Sea Turtles

Turtles such as Loggerheads, live in temperate water. “They inhabit shallow coastal waters, bays, lagoons and even adventure into open waters. The exception is the leatherback turtles, which stays in open waters when not nesting.” (MacInnis, pg. 76) Some turtles migrate long distances; however migration habits defer from species to species.

Sea turtles differ from their land cousins in that they can not retract their head and limbs into their shells. Their flippers have also adapted to assist in swimming. The front flippers serve as a thrust in propulsion, while the back flippers function as a rudder.” (MacInnis, pg. 76) Sea turtles are excellent swimmers, even though they swim...
Due to the sea turtles slow metabolic rate they can dive up to 45 minutes at a time while active. At rest, sea turtles can stay submerged up to five hours, some can even hibernate for months.

Diet varies in sea turtles, some are herbivore, others carnivores and still some eat both plants and animals. Sea turtles eat things such as crustacean, echinoderm, fish and plants.

Due to the sea turtles slow metabolic rate they can dive up to 45 minutes at a time while active. At rest, sea turtles can stay submerged up to five hours, some can even hibernate for months.

Diet varies in sea turtles, some are herbivore, others carnivores and still some eat both plants and animals. Sea turtles eat things such as crustacean, echinoderm, fish and plants.

b. Sea Snakes

Marine sea snakes can be found to reside in both land and the deep ocean waters. They are covered with smooth scales. Some are brightly colored, while others are dull in color. They marine reptiles are equipped with a small head that possesses deadly fangs that release poison when they attack their prey. Once the poison is released, their prey is paralyzed within seconds.

2. Fish: Bony Fish (Osteithchthyes) and Cartilaginous (Chondrichthyes)

Fish are diverse animals, varying in size, shape and habitat. Feeding strategies also differ, allowing many species to co-exist in the same community. Their habitat is determined by the temperature of the ocean waters. Many fish live in tropical, temperate and polar seas. They also find a preferred habitat within each of these zones.

Basic features that are common to all fish is that they have gills throughout their lives and most have fins. Many have scales, though size and shape vary from species to species. All fish secrete mucus from their skin, which helps to block bacteria and fungus and decrease water friction. Many of the fish are colorful, which serves as a mode of protection.
against predators. “Specialized pigments called chromatophore produce most of these colors that help camouflage fish.” (Berrill, pg. 153) These cells are linked to the fishes nervous system, allowing the cells to change in color, especially when danger is sensed.

For survival to occur, fish rely on their five senses. They have highly developed eyes, which function similar to humans. These eyes move independently of each other. This helps to keep predators in view from all angles.

Water transmits sound better than air. The sounds fish make alert them to predators and allow them to communicate. Fish also have a well-developed sense of smell, which is beneficial when hunting for prey. Some are capable of emitting chemicals with distinct odors to communicate with members of the same species. Lastly, fish have a poor sense of taste, but have a good sense of touch.

Due to the diversity of these animals, they have been broken down into two classes of fish, bony and cartilaginous fish. Bony fish have a skeleton made up of bone. In contrast, cartilaginous fish are made up of cartilage.

**a. Bony Fish:** Bony fish have scales that cover their skin and gills on either side of their head that are covered by the operculum. Their mouth is located in the front of the body, along with having symmetry in the shape of their tails. “They also have a swim bladder to control the amount of gas in their system, which enables them to float.” (Berrill, p. 155) This helps bony fish maintain their position in the ocean. Lastly they reproduce by spawning, releasing sperms and eggs in the water simultaneously or males release sperm over the egg.

1.) **Flounder:** Flounder reside along the ocean floor. They easily blend into the sand environment with their special ability to change color. This bony fish is also unique because it has two eyes on one side of its head. As flounder develop one eye migrates to the other side if its head. This helps it to view prey as it lies along the ocean floor.
2.) Tuna: Tuna are of the fastest bony fish, due to its crescent shape. These fish maintain their place in the ocean due to their swim bladder. However, they are known to dive into mid-ocean in search for food.

b. Cartilaginous Fish: Cartilaginous fish have tooth-like structures that are similar to scales. They only have five to seven gill slits and lack an operculum. The mouth of these fish rests below the head and fins are asymmetrical in design. Cartilaginous fish lack a swim bladder, but have the ability float due to higher levels of oils secreted by the liver. Lastly, cartilaginous fish reproduce by internal copulation.

Sharks belong to the scientific group known as elasmobranchs. They inhabit all types of waters. In general, more active sharks are larger and occupy a greater territory. Bottom-dwelling sharks are less active and are usually smaller in size.

“A typical shark has a slender, graceful body that is slightly thicker in the middle and tapers at both ends.” (Miller, pg. 32) This allows for their sleek body movement in ocean waters. They have a pair of dorsal fins along the top of their back and sickle shaped tail whose upper lobe is longer than the lower lobe. The mouth is slightly lower than the head.

Sharks lack a swim bladder, which aides in buoyancy. However, they are efficient swimmers. The cartilaginous skeleton also helps to compensate for the lack of a swim bladder and allows for flexibility and graceful movement. They are able to thrust forward with their powerful tails. In addition, their pectoral fins, located along their sides aide in turning while swimming in the ocean waters. Sharks have enormous livers, which helps to compensate for the lack of swim bladders because it is rich in oil. This is beneficial because oil is less dense than water, making the shark more buoyant.

Along with their superior swimming ability, sharks use a combina-
3. Mammals: Some mammals, such as dolphins and whales, rely on a variety of senses to acquire food. They also rely on senses such as smell, feel, hearing, sight, taste and their special ability to detect electrical and magnetic fields. A keen sense of smell is helpful in seeking food. In addition to this sense, the lateral line runs along the shark’s body to assist it in feeling things, such as vibrations in waves and pressure changes. This is beneficial in detecting low frequency changes, especially since a shark’s sense of hearing is poor. Vision levels vary depending on the shark’s habitat. For example, sharks that live in clear waters rely on sight more than sharks that live in deeper waters.

Lastly, one of the most remarkable features of the shark is their ability to detect electrical fields. “They do this with a special organ called the Ampullae of Lorenzini, which are small gel filled pits in the snout and other parts of the body.” (Miller, 34) All living organisms create electrical fields around them that sharks can detect. This additional sense is beneficial in seeking prey.

1.) Scalloped Hammerhead

Scalloped Hammerhead is known for its unusual design. Its head provides additional lift for the shark’s body, similar to the wings of an airplane. The wider distribution enables hammerhead to locate prey faster. These sharks usually travel in schools, making distinct patterns that are believed to be forms of communication.

2.) Manta-Ray

Manta Rays are known as the “Devil Fish” due to their cephalic-like lobes and horn-like appendages that protrude from their head. Their bodies are triangular in shape and they are surface dwellers. They are equipped with a pair of head fins that extend beyond their eyes and usually carried furled up into tight rolls. When Manta-Rays encounter a swarm of shrimp or small fish, their fins unroll and funnel food toward the mouth. They weigh over three thousand pounds. Rays thrive in tropical areas, preferring open waters.
3. MAMMALS

Mammals are air-breathing warm blooded vertebrates that bear live young, nurse their offspring and have hair or fur at some point during their development.

The water presents unique issues for mammals because it absorbs body heat more rapidly than air, which causes concern for warm-blooded mammals. They struggle to maintain their body temperatures. Some have blubber to help insulate their bodies. This fat also provides energy when food is scarce.

Lastly, marine mammals also have developed reproduction strategies, such as long gestation period. This helps to ensure the survival of well developed young.

a. Dolphins and Porpoises
Dolphins and Porpoises are warm-blooded, air breathing mammals that bear and nurse their young. “These animals excel in marine acrobatics, long distance travel and navigation.” (MacInnis, pg. 173) They are fast swimmers and often leap from the water spinning into the air, then diving back into the ocean waters. Dolphins and Porpoises are considered to be highly intelligent and communicate among themselves. Lastly, they have highly social behaviors and curiosity toward humans.

b. Orca Whale
The orca whale is a small whale, but is much larger than porpoises. It is one of the most powerful and ferocious creatures in the ocean. Orca Whales are known for their towering dorsal fin. Also, the females are known to take on the lead role. These animals inhabit mid-ocean depths.
Conclusion

In testing "Marvin's Marine Adventures," I was unable to acquire many users within the desired age group due to time constraints of the project. Three individuals within the 11–12 year old age group, twenty-five individuals ranged from 19–30 age group and twelve individuals were from 31–55 years in age tested the prototype. Even though the age groups varied from my target audience of 11–12 years old, similar results were discovered within all age groups.

Users were not interested in reading the "Information" section. Some would skim a section or two, but most would not. They appeared to enjoy the increased activity associated with viewing the information section.

Another issue observed was users did not access the "Help" to obtain game play instructions. This behavior was anticipated and the game was designed to be as self-explanatory as possible. Instructions were located at the bottom of the periscope and warning messages were programmed to appear when incorrect buttons were clicked. This was helpful to users, especially for inexperienced users. They were able to play the game successfully after realizing the pattern of the game.

Three areas appeared to be a continual concern in regard to users understanding of "Marvin's Marine Adventures." They had difficulty understanding their location within this information section of the prototype. To clear up this confusion, the level indicator, cursor and the letter "A" on the Menu button were color coded. This helped to organize the different levels of material presented. To become more aware of their orientation within this section users could also click on the "Map" button. This button offered a graphical representation of the different areas, along with highlighting their current location.

Users were also confused in the interactive game. After reading the "Who Am I?" questions, they did not know to select an ocean depth to view marine animals. To avoid further confusion, instructions were added at the bottom of the periscope and warning messages were programmed in the other buttons to encourage users to select the ocean depth slider bar.

Lastly, there was difficulty navigating Marvin to "check" answers. Users did not understand that they needed to intersect Marvin into a marine animal before checking their
Results and Implications of a Trail Implementation

In testing “Marvin’s Marine Adventures,” I was unable to acquire many users within the desired age group due to time constraints of the project. Three individuals within the 11–12 year old age group, twenty-five individuals ranged from 19–30 age group and twelve individuals were from 31–55 years in age tested the prototype. Even though the age groups varied from my target audience of 11–12 years old, similar results were discovered within all age groups.

Users were not interested in reading the “Information” section. Some would skim a section or two, but most went directly to the interactive game. They appeared to enjoy the increased activity and challenges of the game as opposed to viewing the information section.

Another issue observed was users did not access the “Help” to obtain game play instructions. This behavior was anticipated and the game was designed to be as self-explanatory as possible. Instructions were located at the bottom of the periscope and warning messages were programmed to appear when incorrect buttons were clicked. This was helpful to users, especially for inexperienced users. They were able to play the game successfully after realizing the pattern of the game.

Three areas appeared to be a continual concern in regard to users understanding of “Marvin’s Marine Adventures.” They had difficulty understanding their location within this information section of the prototype. To clear up this confusion, the level indicator, cursor and the letter “M” on the Menu button were color coded. This helped to organize the different levels of material presented. To become more aware of their orientation within this section users could also click on the “Map” button. This button offered a graphical representation of the different areas, along with highlighting their current location.

Users were also confused in the interactive game. After reading the “Who Am I?” questions, they did not know to select an ocean depth to view marine animals. To avoid further confusion, instructions were added at the bottom of the periscope and warning messages were programmed in the other buttons to encourage users to select the ocean depth slider bar.

Lastly, there was difficulty navigating Marvin to “check” answers. Users did not understand that they needed to intersect Marvin into a marine animal before checking their
was provided when users intersect Marvin with marine animals. Instructions were also added at the bottom of the periscope.

Another common behavior observed was users did not take advantage of cross referencing material by returning to the information section. They preferred to access the hint button or continue to make arbitrary guesses. Users would rather obtain a quick response from a guess than take the time to reference material. At the current time, changes have not been made because I felt that “Marvin’s Marine Adventures” would need to undergo major reconstruction and design of the game. Unfortunately, time limitations prevented me from making any further changes. However, if I were to continue working on this project I would make several recommendations. For example, I would eliminate the information section at the beginning of “Marvin’s Marine Adventures” and users would begin with the interactive game. I would link the information section to the “Who Am I?” questions. If users click on a word within the question, they would review material of the unknown animal. This information would be similar to the material presented in the information section of the prototype.

I would also like to further develop the “Winner” section of the game. At the current time, Marvin congratulates users with his thumbs-up signal and the screen color cycles. However, I would develop a section in which users create their own fish. They would use marine animal characteristics that they learned while playing the game to create their version of a fish. They would select different characteristics by clicking and dragging animal parts into the creation area. For example, a fish may have fur, a cartilaginous skeleton, flippers and scales on its underside. They would be able name their fish and print the document. I believe this would be a fun way to reinforce the information that was previously presented and enhance user creativity.

In conclusion, I am very pleased with the outcome of “Marvin’s Marine Adventures.” I believe users were able to learn about the eight vertebrate marine animals, work on their memory recall skills and engage in hypothesis testing. However, they did not utilize their cross reference skills while playing “Marvin’s Marine Adventures.” They appeared to remain in the interactive game until they won, lost or quit out of the prototype. Lastly, users appeared to be challenged, along with enjoying themselves while playing “Marvin’s Marine Adventures.”


Appendix A
Interactive Game

```
startMovie
    -- preload cast to improve running time
    -- set colordepth and volume.
    puppetSprite 1, TRUE
    puppetSprite 15, TRUE
    puppetSprite 48, True
    set the colordepth = 8
    set the soundlevel = 7
    preloadcost 532
    preloadcost 533
    preloadcost 535
    -- keep so field doesn't get screwed up!! this field stores
    -- the animal names
    set the textFont of field "animallist" to "serif"
    set the textsize of field "animallist" = 16
    set the textFont of
    set the textSize of
    updateStage

end

call an init so doesn't reset when leave movie
init
    puppetSprite 44, True
    global gCountWrongAnswer, gCountRightAnswer, gReturnDict, gRemainingWords
    set the LocX of sprite 44 = 32
    set the LocY of sprite 44 = 24
    ColorCursorConstraintPeg
    put field "animalstorage" into gRemainingWords
    set gCountWrongAnswer = 0
    set gCountRightAnswer = 0
    set gReturnDict = 0
    if gCountRightAnswer = 0 then
        set the counter of sprite 48 to the number of cast "flashcount"
        updateStage
    end if

to set gNumber or else it won't do the repeat
endsnap

variables gNumber, gName, gRemainingWords
set gNumber = the number of lines in field "animallist"
repeat with K = 1 to gNumber
    set J = random (gNumber)
    set line J of field "animallist" into holder
    set line K of field "animallist" into line J of field "animallist"
end repeat

if field "animallist" = empty then
    wait
end if

updateStage
at "game over"
set line 1 of field "animallist" into gName
set line 1 of field "animallist" into gName
set line 1 of field "animallist" into gName
set line 1 of field "animallist" into gName
set line 1 of field "animallist" into gName
set line 1 of field "animallist" into gName
set line 1 of field "animallist" into gName
set line 1 of field "animallist" into gName
set line 1 of field "animallist" into gName
```

on startMovie
--preload cast to improve running time
--set colordepth and volume
puppetsprite 1, TRUE
PuppetSprite 15, TRUE
puppetsprite 48, True
set the colordepth = 8
set the soundLevel = 7
preloadcast 532
preloadcast 533
preloadcast 535
--Keep so field doesn't get screwed up!! This field stores
--the animal names
set the textFont of field "animallist" to "Geneva"
set the textSize of field "animallist" to 12
set the textFont of field "animalstorage" to "Geneva"
set the textSize of field "animalstorage" to 12
updateStage
end

--call an init so doesn't reset when leave movie
on init
  puppetsprite 44, True
  Global gCountwronganswer, gCountrightanswer, gReturnDict, gRemainingwords
  set the LocH of sprite 44 = 32
  set the LocV of sprite 44 = 24
  ColorCursorConstraintPeg
  put field "animalstorage" into gRemainingwords
  set gCountwronganswer = 0
  set gCountrightanswer = 0
  set gReturnDict = 0
  if gCountrightanswer = 0 then
    set the castnum of sprite 48 to the number of cast "fishiecount"
  updateStage
  end if
end

--need to set gNumber or else it won't do the random
--swaps the lines
on randomswap
  Global gNumber, gName, gRemainingwords
  set gNumber = the number of lines in field "animallist"
  repeat with K = 1 to gNumber
    set J = random (gNumber)
    put line J of field "animallist" into holder
    put line K of field "animallist" into line J of field "animallist"
    put holder into line K of field "animallist"
  end repeat
  if field "animallist" = empty then
    invis
    updateStage
    go "Game Over"
  else
    put line 1 of field "animallist" into gName
    Go gName
    delete line 1 of field "animallist"
    put field "animallist" into gRemainingwords
  end if
end

updateStage
--TO HILITE SUB

On swapsub
  set the castNum of sprite 44 to the number of cast "whitesub"
  updatestage
end

On swapsubback
  set the castNum of sprite 44 to the number of cast "allblack"
  updatestage
end

--to animate on the go the frame use a counter, remember its counting backwards
On AnimateWater
  puppetSprite 9, True
  global gloopcnt
  if (random(1) = 1) and (gloopcnt < 1) then
    set gloopcnt = 2
  end if
  if gloopcnt > 0 then
    set the castNum of Sprite 9 to 502 + gloopcnt
    set gloopcnt = gloopcnt - 1
  end if
  updatestage
end

On AnimateSUB
  puppetSprite 8, True
  global gloopcnts
  --if the puppet of sprite 8 = true then
  if (random(1) = 1) and (gloopcnts < 1) then
    set gloopcnts = 9
  end if
  if gloopcnts > 0 then
    set the castNum of Sprite 8 to 507 + gloopcnts
    set gloopcnts = gloopcnts - 1
  end if
  updatestage
end

--pegmarvin allows marvin to reset to middle on mouseup
--deleted color cursor option, possible add later
On ColorCursorConstraintPeg
  set the mouseupScript to "pegMarvin"
  updatestage
end

--store the animal for the profile section
--put P after so know where to go on score
On Animalprofile
  Global gProfile, gName
  put gName into gProfile
  put "P" after gProfile
end

--allows slider animation to occur..swap cast
--call handler in submarine button
--turn on puppetsprites to see swap of depth
On slider
global gDepth
    set gDepth = the LocV of sprite 44
if (gDepth >= 24) and (gDepth <= 27) then
    set the castNum of sprite 42 to the number of cast "Bar2"
    updatestage
end if
if (gDepth >= 28) and (gDepth <= 46) then
    set the castNum of sprite 42 to the number of cast "Bar3"
    updatestage
end if
if (gDepth >= 47) and (gDepth <= 64) then
    set the castNum of sprite 42 to the number of cast "Bar4"
    updatestage
end if
if (gDepth >= 65) and (gDepth <= 83) then
    set the castNum of sprite 42 to the number of cast "Bar5"
    updatestage
end if
put the LocV of sprite 44 into gDepth
end

On NewDive
    set the LocV of sprite 44 = 26
    set the castNum of sprite 42 to the number of cast "Bar2"
    updatestage
end

---submerge allows you to view animation levels to find animals
--must turn off puppetsprites so they will animate
On Submerge
    puppetSprite 44, False
    puppetSprite 2, True
    Global gDepth, gBgswap
    if (gDepth >= 24) and (gDepth <= 27) then
        put the castNum of sprite 2 into gBgswap
        go "TransDepth4"
    end if
    if (gDepth >= 28) and (gDepth <= 46) then
        put the castNum of sprite 2 into gBgswap
        go "TransDepth1"
    end if
    if (gDepth >= 47) and (gDepth <= 64) then
        put the castNum of sprite 2 into gBgswap
        go "TransDepth2"
    end if
    if (gDepth >= 65) and (gDepth <= 83) then
        put the castNum of sprite 2 into gBgswap
        go "TransDepth3"
    end if
    updatestage
end

--ROLOVERS for BUTTONS
On buttonrolls
    puppetSprite 33, True
if rollover (27) then
  set the castNum of sprite 27 to the number of cast "Quitroll"
else
  set the castNum of sprite 27 to the number of cast "Quit"
  update stage
end if

if rollover (28) then
  set the castNum of sprite 28 to the number of cast "Dictionaryroll"
else
  set the castNum of sprite 28 to the number of cast "Dictionary"
  update stage
end if

if rollover (29) then
  set the castNum of sprite 29 to the number of cast "Hilproll"
else
  set the castNum of sprite 29 to the number of cast "Help"
  update stage
end if

if rollover (30) then
  set the castNum of sprite 30 to the number of cast "Guessroll"
else
  set the castNum of sprite 30 to the number of cast "Guessreg"
  update stage
end if

if rollover (32) then
  set the castNum of sprite 32 to the number of cast "Hintroll"
else
  set the castNum of sprite 32 to the number of cast "Hintreg"
  update stage
end if

if rollover (31) then
  set the castNum of sprite 31 to the number of cast "Inforoll"
else
  set the castNum of sprite 31 to the number of cast "Info"
  update stage
end if

if rollover (33) then
  set the castNum of sprite 33 to the number of cast "Diveroll"
else
  set the castNum of sprite 33 to the number of cast "dive"
  update stage
end if

--Rollovers for hint screen
On buttonrolloshint
  puppetSprite 33, True
  puppetSprite 31, True
  puppetSprite 29, True
  puppetSprite 28, True
  puppetSprite 27, True
  puppetSprite 32, True
  puppetSprite 30, True

if rollover (27) then
  set the castNum of sprite 27 to the number of cast "Quitroll"
else
  set the castNum of sprite 27 to the number of cast "Quit"
  updatestage
end if
if rollover (28) then
  set the castNum of sprite 28 to the number of cast "Dictionaryroll"
else
  set the castNum of sprite 28 to the number of cast "Dictionary"
  updatestage
end if
if rollover (29) then
  set the castNum of sprite 29 to the number of cast "Helproll"
else
  set the castNum of sprite 29 to the number of cast "Help"
  updatestage
end if
if rollover (30) then
  set the castNum of sprite 30 to the number of cast "Guessroll"
else
  set the castNum of sprite 30 to the number of cast "Guessreg"
  updatestage
end if
if rollover (32) then
  set the castNum of sprite 32 to the number of cast "Hintrollback"
else
  set the castNum of sprite 32 to the number of cast "Hintreg"
  updatestage
end if
if rollover (31) then
  set the castNum of sprite 31 to the number of cast "Inforoll"
else
  set the castNum of sprite 31 to the number of cast "Info"
  updatestage
end if
if rollover (33) then
  set the castNum of sprite 33 to the number of cast "Diveroll"
else
  set the castNum of sprite 33 to the number of cast "dive"
  updatestage
end if
end

--Help rollovers
On Helprolls
puppetSprite 33, True
puppetSprite 31, True
puppetSprite 29, True
puppetSprite 28, True
puppetSprite 27, True
puppetSprite 32, True
puppetSprite 30, True

if rollover (27) then
  set the castNum of sprite 27 to the number of cast "Quitroll"
else
  set the castNum of sprite 27 to the number of cast "Quit"
  updatestage
end if
if rollover (28) then
  set the castNum of sprite 28 to the number of cast "Dictionaryroll"
else
  set the castNum of sprite 28 to the number of cast "Dictionary"
  updatestage
if rollover (29) then
  set the castNum of sprite 29 to the number of cast "Helprollstart"
else
  set the castNum of sprite 29 to the number of cast "Help"
update stage
end if

if rollover (30) then
  set the castNum of sprite 30 to the number of cast "Guessroll"
else
  set the castNum of sprite 30 to the number of cast "Guessreg"
update stage
end if

if rollover (32) then
  set the castNum of sprite 32 to the number of cast "Hintroll"
else
  set the castNum of sprite 32 to the number of cast "Hintreg"
update stage
end if

if rollover (31) then
  set the castNum of sprite 31 to the number of cast "Inforoll"
else
  set the castNum of sprite 31 to the number of cast "Info"
update stage
end if

if rollover (33) then
  set the castNum of sprite 33 to the number of cast "Diveroll"
else
  set the castNum of sprite 33 to the number of cast "dive"
update stage
end if

end

--help rollover at start of game, two different help sections
On Helprollstarts
puppetSprite 33, True
puppetSprite 31, True
puppetSprite 29, True
puppetSprite 28, True
puppetSprite 27, True
puppetSprite 32, True
puppetSprite 30, True

if rollover (27) then
  set the castNum of sprite 27 to the number of cast "Quitroll"
else
  set the castNum of sprite 27 to the number of cast "Quit"
update stage
end if

if rollover (28) then
  set the castNum of sprite 28 to the number of cast "Dictionaryroll"
else
  set the castNum of sprite 28 to the number of cast "Dictionary"
update stage
end if

if rollover (29) then
  set the castNum of sprite 29 to the number of cast "Helprollstart"
else
  set the castNum of sprite 29 to the number of cast "Help"
update stage
end if

if rollover (30) then

set the castNum of sprite 30 to the number of cast "Guessroll"
else
set the castNum of sprite 30 to the number of cast "Guessreg"
update stage
end if
if rollover (32) then
set the castNum of sprite 32 to the number of cast "Hintroll"
else
set the castNum of sprite 32 to the number of cast "Hintreg"
update stage
end if
if rollover (31) then
set the castNum of sprite 31 to the number of cast "Inforoll"
else
set the castNum of sprite 31 to the number of cast "Info"
update stage
end if
if rollover (33) then
set the castNum of sprite 33 to the number of cast "Diveroll"
else
set the castNum of sprite 33 to the number of cast "dive"
update stage
end if
end
--To show rollover for sub
On SubRollover
puppetSprite 43, True
if rollover (43) then
set the castNum of sprite 43 to the number of cast "OceanDepth"
else
set the castNum of sprite 43 to the number of cast "Bar1"
update stage
end if
end
-- TO GO TO DICTIONARY MOVIE
--to enable the icon and score to stay counted when leave movie
--need to reset constraint and pegMarvin or else lose it
On ReturnDictionary
Global gCountrightanswer, gReturnDict, gRemainingwords
ColorCursorConstraintPeg
------------------------------------------
put gRemainingwords into field "animal1ist"
------------------------------------------
put gCountrightanswer into gReturnDict
puppetSprite 42, True
puppetSprite 44, True
slider
if gReturnDict = 0 then
put gRemainingwords into field "animal1ist"
end if
if gReturnDict = 1 then
set the castNum of sprite 48 to the number of cast "F1"
update stage
end if
if gReturnDict = 2 then
set the castNum of sprite 48 to the number of cast "F2"
update stage
end if
if gReturnDict = 3 then
set the castNum of sprite 48 to the number of cast "F3"
updatestage
end if

if gReturnDict = 4 then
  set the castNum of sprite 48 to the number of cast "F4"
  updatestage
end if

if gReturnDict = 5 then
  set the castNum of sprite 48 to the number of cast "F5"
  updatestage
end if

end

---TO CALL HANDLER WHEN LEAVE MOVIES(DICTIONARY/HELP/INFO---

On OutMovies
  Global gMMA, gCountrighanswer, gReturnDict
  put the frame into gMMA
  put gCountrighanswer into gReturnDict
  updatestage
end

--sets marvin invisible
On invis
  set the visible of Sprite 15 to 0
  updatestage
end

--sets marvin visible
on visi
  set the visible of Sprite 15 to 1
  updatestage
end

--To allow marvin to return to the center of periscope on mouseup
on PegMarvin

  --set the location
  set x = the locH of Sprite 15
  set y = the locV of Sprite 15

  --change in Marvin to the center of the screen
  set steps = 20
  set dx = (400 - x) / steps
  set dy = (280 - y) / steps

  --repeat with 1step bounce
  repeat with i = 1 to steps
    set the locH of Sprite 15 = (i*dx) + x
    set the locV of Sprite 15 = (i*dy) + y
    updatestage
  end repeat
end

--ANIMATION POTENTIAL of Marvins swimming
--don't forget animation sequence goes in reverse
--need different marvins at each ocean depth due to anti-aliasing bg
On AnimateMarvin
  global gloocpcnt
  if the locH of Sprite 15 >= 400 then
    if (random(1) = 1) and (gloocpcnt < 1) then
      set gloocpcnt = 7
    end if
  end if

if gloopcnt > 0 then
    set the castNum of Sprite 15 to 351 + gloopcnt
    set gloopcnt = gloopcnt - 1
end if

updatestage
else
    if (random(1) = 1) and (gloopcnt < 1) then
        set gloopcnt = 7
    end if

    if gloopcnt > 0 then
        set the castNum of Sprite 15 to 378 + gloopcnt
        set gloopcnt = gloopcnt - 1
    end if

updatestage
else
    if (random(1) = 1) and (gloopcnt < 1) then
        set gloopcnt = 7
    end if

    if gloopcnt > 0 then
        set the castNum of Sprite 15 to 387 + gloopcnt
        set gloopcnt = gloopcnt - 1
    end if

updatestage
end if

On AnimateMarvinMid

global gloopcnt

if the LocH of Sprite 15 >= 400 then
    if (random(1) = 1) and (gloopcnt < 1) then
        set gloopcnt = 7
    end if

    if gloopcnt > 0 then
        set the castNum of Sprite 15 to 360 + gloopcnt
        set gloopcnt = gloopcnt - 1
    end if

updatestage
else
    if (random(1) = 1) and (gloopcnt < 1) then
        set gloopcnt = 7
    end if

    if gloopcnt > 0 then
        set the castNum of Sprite 15 to 396 + gloopcnt
        set gloopcnt = gloopcnt - 1
    end if

updatestage
end if

On AnimateMarvinDown

global gloopcnt

if the LocH of Sprite 15 >= 400 then
    if (random(1) = 1) and (gloopcnt < 1) then
        set gloopcnt = 7
    end if

    if gloopcnt > 0 then
        set the castNum of Sprite 15 to 369 + gloopcnt
        set gloopcnt = gloopcnt - 1
    end if

updatestage
else
    if (random(1) = 1) and (gloopcnt < 1) then
        set gloopcnt = 7
    end if

    if gloopcnt > 0 then
        set the castNum of Sprite 15 to 396 + gloopcnt
        set gloopcnt = gloopcnt - 1
    end if

updatestage
end if
end if
update stage
end if
if gGuessed = "no" then
if Sprite 15 intersects 24 then
put "Hammerhead" into gGuess
puppetSound "Chris-garggle-boy"
update stage
end if
if Sprite 15 intersects 17 then
put "Tuna" into gGuess
puppetSound "Chris-garggle-boy"
update stage
end if
if Sprite 15 intersects 19 then
put "Dolphin" into gGuess
puppetSound "Chris-garggle-boy"
update stage
end if
if Sprite 15 intersects 21 then
put "Mantaray" into gGuess
puppetSound "Chris-garggle-boy"
update stage
end if
end if
end if
--------POTENTIAL GUESS--------
--FOR this section to work at three different levels it is necessary to create
--three different handlers or it will read channels in previous sections
On potentialguessUp
global gGuess, gGuessed
if gGuessed = "no" then
if Sprite 15 intersects 17 then
put "Tuna" into gGuess
puppetSound "Chris-garggle-boy"
update stage
end if
if Sprite 15 intersects 20 then
put "Killerwhale" into gGuess
puppetSound "Chris-garggle-boy"
update stage
end if
if Sprite 15 intersects 22 then
put "Turtle" into gGuess
puppetSound "Chris-garggle-boy"
update stage
end if
end if
end
On potentialGuessMid
global gGuess, gGuessed
if gGuessed = "no" then
if Sprite 15 intersects 17 then
put "Tuna" into gGuess
puppetSound "Chris-garggle-boy"
update stage
end if
if Sprite 15 intersects 20 then
put "Killerwhale" into gGuess
puppetSound "Chris-garggle-boy"
update stage
end if
if Sprite 15 intersects 22 then
put "Turtle" into gGuess
puppetSound "Chris-garggle-boy"
update stage
end if
end if
on potentialGuessDown
  global gGuess, gGuessed
  if gGuessed = "no" then
    if Sprite 15 intersects 16 then
      put "SeaSnake" into gGuess
      puppetSound "Chris-garggle-boy"
      updatestage
    end if
    if Sprite 15 intersects 23 then
      put "Flounder" into gGuess
      puppetSound "Chris-garggle-boy"
      updatestage
    end if
  end if
end

Score Script11
on exitFrame
  init
  put field "animalstorage" into field "animalslist"
  updatestage
end

Score Script12
on exitFrame
  Global gProfile
  puppetSprite 11, True
  buttonrolls
  ReturnDictionary
  set the castNum of sprite 11 to the number of cast gProfile
  updatestage
  go the frame
end
Score Script14

on exitFrame
  ColorCursorConstraintPeg
  AnimateMarvin
  SubRollOver
  buttonRolls
  ReturnDictionary
  go the frame
end

Score Script16

on exitFrame
  puppetSprite 8, False
  puppetSprite 9, False
  visit
  ColorCursorConstraintPeg
  AnimateMarvin
  HelpRolls
  ReturnDictionary
  go the frame
end

Score Script18

on exitFrame
  HelpRolls
  returnDictionary
  go the frame
end

Score Script20

on exitFrame
  ColorCursorConstraintPeg
  AnimateMarvin
end
on exitFrame
    Global gCountwronganswer, gCorrectanswer
    --to keep marvin within the bounding box--
    --so he does not look like he is swimming off the screen
    puppetSound "possible behindquestions"
    ColorCursorConstraintPeg
    AnimateMarvin
    if gCorrectanswer >= 5 then init
    set gCountwronganswer = 0
    swapsubback
    updatestage
end

---

on mouseUp
end

---

on exitFrame
end

---

on mouseUp
end

---

on exitFrame
end
Score Script26

on exitFrame
   visi
   ColorCursorConstraintPeg
   AnimateMarvin
   Helprollstarts
   ReturnDictionary
   go the frame
end

Score Script27

on exitFrame
   buttonrolls
   delay 10
end

Score Script28

on exitFrame
   delay 60
   go "Turtle"
end

Score Script29

on enterFrame
   buttonrolls
   ColorCursorConstraintPeg
   AnimateMarvin
   ReturnDictionary
   swapsub
end

on exitFrame
   go the frame
end
Script of Cast Member 45: warning

on mouseUp
  puppetSprite 36, False
end puppetSprite 37, False

repeat while the stillDown
  set the castnum of sprite 31 to the number of cast "InfoReverse"
  updateStage
end repeat

Script of Cast Member 46: warning messageblock

on mouseUp
  busy(2) then sound stop 2
  busy(3) then sound stop 3
  sound "click" set the castnum of sprite 31 to the number of cast "InfoReverse"
  updateStage
end

Script of Cast Member 47: Warning2

on mouseUp
  puppetSprite 14, True
  set the castnum of sprite 14 to the number of cast "Warning subindexed"
  sound playfile 3, "real little buttons"
  updateStage
  set the timer = 0
  repeat while the timer < 5 * 60
    end repeat
  set the castnum of sprite 14 to the number of cast "warning"
  updateStage
  puppetSprite 14, False
end

ReturnDictionary

Script of Cast Member 48: warning continue

on mouseUp
  puppetSprite 14, True
  set the castnum of sprite 14 to the number of cast "Warning continue indexed"
  sound playfile 3, "real little buttons"
  updateStage
  set the timer = 0
  repeat while the timer < 5 * 60
    end repeat
  set the castnum of sprite 14 to the number of cast "warning"
  updateStage
  puppetSprite 14, False
end
Script of Cast Member 71: Inforoll

on mousedown
  puppetSprite 36, False
  puppetSprite 37, False
  repeat while the stilldown
    set the castNum of sprite 31 to the number of cast "Infoinverse"
    updatestage
  end repeat
end

on mouseUp
  if soundbusy(2) then sound stop 2
  if soundbusy(5) then sound stop 5
  if soundbusy(6) then sound stop 6
  puppetsound "click"
  set the castNum of sprite 31 to the number of cast "Inforoll"
  updatestage
OutMovies
  go to frame "Loadinfo" of movie "Pretestinfo"
end

Score Script 76

on exitFrame
  buttonrolls
  AnimateMarvin
  potentialGuessUp
  ReturnDictionary
end

Score Script 79

on exitFrame
  buttonrolls
  potentialGuessUp
  ReturnDictionary
  go "animateUp"
end
on mouseDown
    puppetSprite 34, True
    repeat while the stilldown
        set the castNum of sprite 34 to the number of cast "NoInversequit"
        updateStage
    end repeat
end

on mouseUp
    Global gQuit
    puppetSprite 8, False
    puppetSprite 9, False
    set the castNum of sprite 34 to the number of cast "NoQuit"
    sound playfile 4, "Mouse Down"
    --to return back to section if don't quit
    updateStage
    puppetSprite 34, False
    puppetTransition 51, 4, 20
    if (gQuit >= 29 and gQuit <= 56) then
        go "Dolphin"
    end if
    if (gQuit >= 62 and gQuit <= 83) then
        go "Flounder"
    end if
    if (gQuit >= 89 and gQuit <= 110) then
        go "Killerwhale"
    end if
    if (gQuit >= 115 and gQuit <= 147) then
        go "Turtle"
    end if
    if (gQuit >= 153 and gQuit <= 194) then
        go "Hammerhead"
    end if
    if (gQuit >= 200 and gQuit <= 220) then
        go "Tuna"
    end if
    if (gQuit >= 226 and gQuit <= 264) then
        go "Mantaray"
    end if
    if (gQuit >= 268 and gQuit <= 306) then
        go "SeaSnake"
    end if
    if (gQuit >= 1 and gQuit <= 28) then
        go gQuit
    end if
    if gQuit >= 307 then
        go gQuit
    end if
    updateStage
end
on exitFrame
  Global gCountWrongAnswer, gCountRightAnswer
  --to keep marvin within the bounding box--
  --so he does not look like he is swimming off the screen
  set the visible of sprite 44 to true
  ColorCursorConstraintPeg
  AnimateMarvin
  SubRollover
  buttonrolls
  ReturnDictionary
  if gCountRightAnswer >= 5 then init
  set gCountWrongAnswer = 0
  swapSubback
  updateStage
  randomSwap
end

Score Script118

on exitFrame
  updateStage
end

Script of Cast Member119

on mouseUp
  go "start"
end
Score Script201

on exitFrame
  sound playfile 5, "watersound"
  buttonrolls
  AnimateMarvin
  potentialguessUp
  ReturnDictionary
end

Score Script202

on exitFrame
  sound playfile 5, "watersound"
  buttonrolls
  AnimateMarvinMid
  potentialGuessMid
  ReturnDictionary
end

Score Script203

on exitFrame
  set the castNum of sprite 2 to the number of cast "Level bg (3000ft)INDEX"
  updateStage
end

Score Script204

on exitFrame
  set the castNum of sprite 2 to the number of cast "Level Bg. (1000 ft.)INDEXED"
  updateStage
end

Script of Cast Member207:Dictionaryinverse

on mouseUp
  OutMovies
  go to frame "Dictionaryterms" of movie "Dictionary Movie"
end
Score Script219

on exitFrame
  global gTrans
  puppetSprite 2, False
  buttonrolls
  puppetTransition 51
  go gTrans
end

Score Script220

on exitFrame
  set the castNum of sprite 2 to the number of cast "bg sand indexed darker"
  updateStage
end

Score Script221

on exitFrame
  go the frame
end

Score Script226

on enterFrame
  sound playfile 5, "watersound"
  updateStage
  ColorCursorConstraintPeg
  buttonrolls
  AnimateMarvin
  potentialguessUp
  ReturnDictionary
end
Script of Cast Member 231: whitesub

on mouseDown
    repeat while the stilldown
        swap sub
        set YLoc = the mouseV
        if YLoc >= 83 then set YLoc = 83
        if YLoc <= 24 then set YLoc = 24
        set the LocV of sprite 44 = YLoc
        slider
        update stage
    end repeat
end

on mouseUp
    puppetsound 0
    swap sub back
    submerge
end

Score Script 232

on exitFrame
    set the castNum of sprite 44 to the number of cast "whitesub"
    update stage
end

Start ANSWER------

Score Script 233

on exitFrame
    set the castNum of sprite 44 to the number of cast "allblack"
    update stage
end
on mousedown
  repeat while the stilldown
    set the castNum of sprite 30 to the number of cast "Guessinverse"
  end repeat
end

on mouseUp
  -- To determine if the guess = the question
  -- gName is from the random selection of questions
  -- gGuess is the correct answer
  -- gWrongAnswer warns the user and takes them to help screens
  -- gScreen put the frame number into memory so can go back after get hint
  puppetsound 0
  ---- RIGHT ANSWER ----
  global gName, gGuess, gCountWrongAnswer, gHint, gScreen, gCountRightAnswer, gDepth, gGuessed, gBgSwap
  puppetSprite 42 True
  puppetSprite 44, True
  updateStage
  set gGuessed = "no"
  slider
  if (gName = gGuess) then
    set gCountRightAnswer = gCountRightAnswer + 1
  set the castNum of sprite 30 to the number of cast "Guessroll"
  puppetsound "click"
  updateStage
  if (gCountWrongAnswer < 1) and (gCountRightAnswer = 1) then
    set the castNum of sprite 48 to the number of cast "F1"
    AnimalProfile
    invis
    if soundBusy(2) then sound stop 2
    if soundBusy(5) then sound stop 5
    if soundBusy(6) then sound stop 6
    updateStage
    go "right"
    sound playFile 7, "Right answer beep"
    updateStage
  end if
  if (gCountWrongAnswer < 1) and (gCountRightAnswer = 2) then
    set the castNum of sprite 48 to the number of cast "F2"
    AnimalProfile
    invis
    if soundBusy(2) then sound stop 2
    if soundBusy(5) then sound stop 5
    if soundBusy(6) then sound stop 6
    updateStage
    go "right"
    sound playFile 7, "Right answer beep"
    updateStage
  end if
  if (gCountWrongAnswer < 1) and (gCountRightAnswer = 3) then
    set the castNum of sprite 48 to the number of cast "F3"
    AnimalProfile
    invis
    sound "click"
  end if

if soundbusy(2) then sound stop 2
if soundbusy(5) then sound stop 5
if soundbusy(6) then sound stop 6
updatestage
go "right"
sound playfile 7, "Right answer beep"
end if

if (gCountwronganswer < 1) and (gCountrightanswer = 4) then
set the castNum of sprite 48 to the number of cast "F4"
Animalprofile
invisi
if soundbusy(2) then sound stop 2
if soundbusy(5) then sound stop 5
if soundbusy(6) then sound stop 6
updatestage
go "right"
sound playfile 7, "Right answer beep"
end if

if (gCountwronganswer < 1) and (gCountrightanswer = 5) then
set the castNum of sprite 48 to the number of cast "F5"
Animalprofile
invisi
updatestage
set the timer = 0
repeat while the timer < 1 * 60
end repeat
if soundbusy(2) then sound stop 2
if soundbusy(5) then sound stop 5
if soundbusy(6) then sound stop 6
updatestage
go "Winner"
end if
put gCountrightanswer
end if

----WRONG ANSWER----
--gname and gguess do not = each other then wrong answer
--I use a counter to keep track of score
--use timer to flash wrong answer warning 1st attempt
--2nd attempt use hint button
--3rd attempt wrong answer screen

if (gName <> gGuess) then
puppetSprite 14, True
set gCountwronganswer = gCountwronganswer + 1
if gCountwronganswer = 1 then
set the castNum of sprite 14 to the number of cast "warningwrongone"
set the castNum of sprite 30 to the number of cast "Guessroll"
sound playfile 3, "real little buttons"
updatestage
set the timer = 0
repeat while the timer < 5 * 60
end repeat
set the castNum of sprite 14 to the number of cast "Swapwrongone"
updatestage
puppetSprite 14, False
end if

if gCountwronganswer = 2 then
set the castNum of sprite 30 to the number of cast "Guessroll"
puppetsound "click"
update stage
  Put the frame into gScreen
  put gName into ghInt
  put "s" after ghInt
  set the castNum of sprite 2 to gBgswap
  if soundbusy(2) then sound stop 2
  if soundbusy(5) then sound stop 5
  if soundbusy(6) then sound stop 6
  update stage
  puppetTransition 51, 4, 20
  go ghInt
  end if
  if gCountwronganswer = 3 then
    set the castNum of sprite 30 to the number of cast "Guessroll"
    puppetSound "click"
  update stage
  Put the frame into gScreen
  put gName into gMissed
  put "miss" after gMissed
  Animalprofile
  invis
  if soundbusy(2) then sound stop 2
  if soundbusy(5) then sound stop 5
  if soundbusy(6) then sound stop 6
  update stage
  go gMissed
  end if
end if

----WRONG AND THEN RIGHT ANSWER----
--keeps track of wrong then right
--sends user to right without fish icon
--keeps track with counter
--turn off sounds with soundbusy--works great!

if (gCountwronganswer = 1) and (gName = gGuess) then
  set gCountrightanswer = gCountrightanswer - 1
  Animalprofile
  update stage
  invis
  if soundbusy(2) then sound stop 2
  if soundbusy(5) then sound stop 5
  if soundbusy(6) then sound stop 6
  update stage
  go "right"
end if

if (gCountwronganswer = 2) and (gName = gGuess) then
  set gCountrightanswer = gCountrightanswer - 1
  Animalprofile
  update stage
  invis
  if soundbusy(2) then sound stop 2
  if soundbusy(5) then sound stop 5
  if soundbusy(6) then sound stop 6
  update stage
  go "right"
end if
end
on mouseDown
    --to go to a hint screen by using the button
    puppetSprite 42 True
    puppetSprite 44, True
    repeat while the stilldown
        set the castNum of sprite 32 to the number of cast "Hintinverse"
        updateStage
    end repeat
end

on mouseUp
    --add but, for frame marker label
    --gScreen allows to keep frame in container for return
    if soundbusy(2) then sound stop 2
    if soundbusy(5) then sound stop 5
    if soundbusy(6) then sound stop 6
    Global gScreen, gButtonHint, gName
    puppetsound "click"
    set the castNum of sprite 32 to the number of cast "Hintroll"
    updateStage
    put the frame into gScreen
    put gName into gButtonHint
    put "but" after gButtonHint
    slider updateStage
    puppetTransition 51, 2, 20
    go gButtonHint
end
on mousedown
    repeat while the stilldown
        set the castNum of sprite 29 to the number of cast "Help inverse"
        updatestage
    end repeat
end

on mouseUp
    if soundbusy(2) then sound stop 2
    if soundbusy(5) then sound stop 5
    if soundbusy(6) then sound stop 6
    puppetsound "click"
    set the castNum of sprite 29 to the number of cast "Helproll"
    put the frame into gHelp
    updatestage
    puppetTransition 51, 4, 20
    go "Help"
end
Script of Cast Member243: Quitroll

on mouseDown
  repeat while the stilldown
    set the castNum of sprite 27 to the number of cast "Quitinverse"
    updateStage
  end repeat
end

on mouseUp
  if soundbusy(2) then sound stop 2
  if soundbusy(5) then sound stop 5
  if soundbusy(6) then sound stop 6
  Global gQuit
  puppetsound "click" I need to be specific and send to start
  set the castNum of sprite 27 to the number of cast "Quitroll"
  updateStage
  put the frame into gQuit
  invisible puppetsound "click"
  puppetTransition 51, 4, 20 to the number of cast "Quitroll"
  go "Quit"
end

Script of Cast Member245

on mouseUp
end
on mouseDown
--to go to a hint screen by using the button
puppetSprite 42 True
puppetSprite 44, True
repeat while the stilldown
    set the castNum of sprite 32 to the number of cast "Hintinverse"
    updatestage
end repeat
end

on mouseUp
--the sound resets, I need to be specific and send to start
--button on hint screen to return users to previous location
--numbers are frame numbers, within in a location
Global gScreen
puppetsound "click"
set the castNum of sprite 32 to the number of cast "Hintroll"
updatestage
puppetTransition 51, 2, 20
if (gScreen >= 29 and gScreen <= 56) then
    go "Dolphins"
end if
if (gScreen >= 62 and gScreen <= 83) then
    go "Flounders"
end if
if (gScreen >= 89 and gScreen <= 110) then
    go "Killerwhale"
end if
if (gScreen >= 115 and gScreen <= 147) then
    go "Turtles"
end if
if (gScreen >= 153 and gScreen <= 194) then
    go "Hammerhead"
end if
if (gScreen >= 200 and gScreen <= 220) then
    go "Tuna"
end if
if (gScreen >= 226 and gScreen <= 264) then
    go "Mantaray"
end if
if (gScreen >= 268 and gScreen <= 306) then
    go "SeaSnake"
end if
if (gScreen >= 1 and gScreen <= 28) then
    go gScreen
end if
if gScreen >= 307 then
    go gScreen
end if
end
Script of Cast Member 329

on mouseUp
end

Script of Cast Member 340

on mouseUp
end

Script of Cast Member 352: Upperright

On MouseDown
Global gGuessed
set gGuessed = "no"
end

on mouseUp
Global gGuessed
set gGuessed = "Yes"
end

Script of Cast Member 353

On MouseDown
Global gGuessed
set gGuessed = "no"
end

on mouseUp
Global gGuessed
set gGuessed = "Yes"
end
Information Section
On startMovie
--To set the color depth and sound level
puppetSprite 45, True
set the colordepth = 8
set the soundlevel = 7
preloadcast 481, 485
end

On init
--to put something in global, so goes to frame "LoadMMA"
--this prevents resetting game
Global gMMA
set gMMA = "LoadMMA"
end

---To Change Cursors for levels
--yellow = vertebrate
--orange = cart, bony, reptile, or mammal
--red = species
--need to use resedit--color cursor resource
On Colorcursoryellow
put ColorCursor (mNew) into mycursor
mycursor(mGetSetCursor,209) to the number of cast "Mollusk"
end

On Colorcursorsorange
put ColorCursor (mNew) into mycursor
mycursor(mGetSetCursor,210) to the number of cast "Mammal"
end

On Colorcursored
put ColorCursor (mNew) into mycursor
mycursor(mGetSetCursor,211) to the number of cast "Mammal"
end

On Colorcursorbblack
put ColorCursor (mNew) into mycursor
mycursor(mGetSetCursor,212) to the number of cast "Mammal"
end

--Button Rollovers
On buttonrolls
puppetSprite 35, True
puppetSprite 33, True
puppetSprite 29, True
puppetSprite 28, True
puppetSprite 27, True
puppetSprite 32, True
puppetSprite 30, True

if rollover (27) then
set the castNum of sprite 27 to the number of cast "Dictionaryroll"
else
set the castNum of sprite 27 to the number of cast "Quit"
end if

if rollover (28) then
set the castNum of sprite 28 to the number of cast "Dictionaryroll"
else
    set the castNum of sprite 28 to the number of cast "Dictionary"
end if
if rollover (29) then
    set the castNum of sprite 29 to the number of cast "Helproll"
else
    set the castNum of sprite 29 to the number of cast "Help"
end if
if rollover (30) then
    set the castNum of sprite 30 to the number of cast "Guessroll"
else
    set the castNum of sprite 30 to the number of cast "Guessreg"
end if
if rollover (32) then
    set the castNum of sprite 32 to the number of cast "Map"
else
    set the castNum of sprite 32 to the number of cast "MapT"
end if
if rollover (33) then
    set the castNum of sprite 33 to the number of cast "Diveroll"
else
    set the castNum of sprite 33 to the number of cast "dive"
end if
if rollover (35) then
    set the castNum of sprite 35 to the number of cast "MMAroll"
else
    set the castNum of sprite 35 to 123
end if
if rollover (45) then
    set the castNum of sprite 45 to the number of cast "MenuT"
end if
-- Help buttons, established to for the return to screen globals
On buttonRollsHelp
    puppetSprite 35, True
    puppetSprite 33, True
    puppetSprite 29, True
    puppetSprite 28, True
    puppetSprite 27, True
    puppetSprite 32, True
    puppetSprite 30, True
if rollover (27) then
    set the castNum of sprite 27 to the number of cast "Quitroll"
else
    set the castNum of sprite 27 to the number of cast "Quit"
end if
if rollover (28) then
    set the castNum of sprite 28 to the number of cast "Dictionaryroll"
else
    set the castNum of sprite 28 to the number of cast "Dictionary"
end if
if rollover (29) then
    set the castNum of sprite 29 to the number of cast "HelprollHelp"
else
    set the castNum of sprite 29 to the number of cast "Help"
    updatestage
end if
if rollover (45) then
    set the castNum of sprite 45 to the number of cast "RedMenu" menu
else
    set the castNum of sprite 45 to the number of cast "RedMenu"
    updatestage
end if
if rollover (30) then
    set the castNum of sprite 30 to the number of cast "Guessroll"
else
    set the castNum of sprite 30 to the number of cast "Guessreg"
    updatestage
end if
if rollover (32) then
    set the castNum of sprite 32 to the number of cast "MapT"
else
    set the castNum of sprite 32 to the number of cast "Map"
    updatestage
end if
if rollover (33) then
    set the castNum of sprite 33 to the number of cast "Diveroll"
else
    set the castNum of sprite 33 to the number of cast "dive"
    updatestage
end if
if rollover (35) then
    set the castNum of sprite 35 to the number of cast "MMAroll"
else
    set the castNum of sprite 35 to 123
    updatestage
end if
end rollover (45) then
    set the castNum of sprite 45 to the number of cast "RedBonymenu"
    updatestage
On greyRoll
    if rollover (45) then
        set the castNum of sprite 45 to the number of cast "MenuT"
    else
        set the castNum of sprite 45 to the number of cast "MenuReg"
        updatestage
    end if
On YellowRoll
    if rollover (45) then
        set the castNum of sprite 45 to the number of cast "YellowTmenu"
        updatestage
    else
        set the castNum of sprite 45 to the number of cast "YellowMenu"
        updatestage
    end if
On OrangeRoll
    if rollover (45) then
        set the castNum of sprite 45 to the number of cast "OrangeTmenu"
    else
        set the castNum of sprite 45 to the number of cast "OrangeMenu"
        updatestage
    end if
end
--I needed to add specific rollovers for this level because the map
--causes it to crash with global variable
On RedRollMammal
if rollover (45) then
    set the castNum of sprite 45 to the number of cast "RedTMammalmenu"
else
    set the castNum of sprite 45 to the number of cast "RedMenu"
update stage
end if
end

On RedRollReptile
if rollover (45) then
    set the castNum of sprite 45 to the number of cast "RedTReptilemenu"
else
    set the castNum of sprite 45 to the number of cast "RedMenu"
update stage
end if
end

On RedRollCart
if rollover (45) then
    set the castNum of sprite 45 to the number of cast "RedTCartmenu"
else
    set the castNum of sprite 45 to the number of cast "RedMenu"
update stage
end if
end

On RedRollBony
if rollover (45) then
    set the castNum of sprite 45 to the number of cast "RedTBonymenu"
else
    set the castNum of sprite 45 to the number of cast "RedMenu"
update stage
end if
end

On level
puppetSprite 43, True
if rollover (43) then
    set the castNum of sprite 43 to the number of cast "level indicator icon"
else
    set the castNum of sprite 43 to the number of cast "blacklevel"
update stage
puppetsprite 43, False
end if

----- Rollovers for animal buttons: view specific screens

On Hammerroll
puppetSprite 36, True
if rollover (36) then
    set the castNum of sprite 36 to the number of cast "Hroll"
else
    set the castNum of sprite 36 to the number of cast "HammerH"
update stage
end if
end

On Mantaroll
puppetSprite 37, True
if rollover (37) then

set the castNum of sprite 37 to the number of cast "Mroll"
else
set the castNum of sprite 37 to the number of cast "MantaR"
update stage
end if
end

On Turtleroll
puppetSprite 36, True
if rollover (36) then
set the castNum of sprite 36 to the number of cast "SeaTroll"
else
set the castNum of sprite 36 to the number of cast "SeaT"
update stage
end if
end

On Snakeroll
puppetSprite 37, True
if rollover (37) then
set the castNum of sprite 37 to the number of cast "SeaSroll"
else
set the castNum of sprite 37 to the number of cast "SeaS"
update stage
end if
end

On Orca roll
puppetSprite 36, True
if rollover (36) then
set the castNum of sprite 36 to the number of cast "OrcaWroll"
else
set the castNum of sprite 36 to the number of cast "OrcaW"
update stage
end if
end

On Dolphinroll
puppetSprite 37, True
if rollover (37) then
set the castNum of sprite 37 to the number of cast "Dolfinroll"
else
set the castNum of sprite 37 to the number of cast "Dol"
update stage
end if
end

On Tunaroll
puppetSprite 36, True
if rollover (36) then
set the castNum of sprite 36 to the number of cast "Tunaroll"
else
set the castNum of sprite 36 to the number of cast "Tun"
update stage
end if
end

On Flounderroll
puppetSprite 37, True
if rollover (37) then
set the castNum of sprite 37 to the number of cast "Flounderroll"
else
set the castNum of sprite 37 to the number of cast "Flo"
updatestage
end if
end

---to allow for map functionality
--need to turn on puppet of all squares
--choose range of frames

On mapaction
Global gMap
puppetSprite 4, True
puppetSprite 5, True
puppetSprite 6, True
puppetSprite 7, True
puppetSprite 8, True
puppetSprite 9, True
puppetSprite 10, True
puppetSprite 11, True
puppetSprite 12, True
puppetSprite 13, True
puppetSprite 14, True
puppetSprite 15, True
puppetSprite 16, True

--VERT
if (gMap >= 122 and gMap <= 150) then
  set the castNum of sprite 16 to the number of cast "Mapvertinvert"
  set the castNum of sprite 43 to the number of cast "levelvert"
  updatestage
end if

--REP
if (gMap >= 157 and gMap <= 184) then
  set the castNum of sprite 15 to the number of cast "Maprepinvert"
  set the castNum of sprite 43 to the number of cast "levelrep"
  updatestage
end if

--MAMMALS
if (gMap >= 434 and gMap <= 464) then
  set the castNum of sprite 14 to the number of cast "Mapmaminvert"
  set the castNum of sprite 43 to the number of cast "levelmam"
  updatestage
end if

--BONY FISH
if (gMap >= 250 and gMap <= 272) then
  set the castNum of sprite 13 to the number of cast "Mapboninvert"
  set the castNum of sprite 43 to the number of cast "levelbon"
  updatestage
end if

--CARTILAGINOUS
if (gMap >= 338 and gMap <= 368) then
  set the castNum of sprite 12 to the number of cast "Mapcarinvert"
  set the castNum of sprite 43 to the number of cast "levelcar"
  updatestage
end if

--TURTLES
if (gMap >= 187 and gMap <= 213) then
  set the castNum of sprite 11 to the number of cast "Mapturinvert"
  set the castNum of sprite 43 to the number of cast "leveltur"
updateStage
end if

-- SEA SNAKE
if (gMap >= 217 and gMap <= 244) then
  set the castNum of sprite 10 to the number of cast "MapsnaInvert"
  set the castNum of sprite 43 to the number of cast "levelsna"
  updateStage
end if

-- TUNA
if (gMap >= 275 and gMap <= 301) then
  set the castNum of sprite 6 to the number of cast "MaptunInvert"
  set the castNum of sprite 43 to the number of cast "leveltun"
  updateStage
end if

-- FLOUNDER
if (gMap >= 305 and gMap <= 331) then
  set the castNum of sprite 7 to the number of cast "MapfloInvert"
  set the castNum of sprite 43 to the number of cast "levelflo"
  updateStage
end if

-- HAMMERHEAD
if (gMap >= 370 and gMap <= 396) then
  set the castNum of sprite 5 to the number of cast "MapshaInvert"
  set the castNum of sprite 43 to the number of cast "levelsha"
  updateStage
end if

-- MANTA
if (gMap >= 405 and gMap <= 431) then
  set the castNum of sprite 4 to the number of cast "MaprayInvert"
  set the castNum of sprite 43 to the number of cast "levelman"
  updateStage
end if

-- ORCA
if (gMap >= 470 and gMap <= 496) then
  set the castNum of sprite 9 to the number of cast "MapwhaInvert"
  set the castNum of sprite 43 to the number of cast "levelwha"
  updateStage
end if

-- DOLPHIN
if (gMap >= 500 and gMap <= 526) then
  set the castNum of sprite 8 to the number of cast "MapdolInvert"
  set the castNum of sprite 43 to the number of cast "leveldol"
  updateStage
end if
end

-- to animate on the go the frame
On AnimateWater
  puppetSprite 9, True
  global gloopcnt

  if (random(1) = 1) and (gloopcnt < 1) then
    set gloopcnt = 2
  end if

  if gloopcnt > 0 then
    set the castNum of Sprite 9 to 192 + gloopcnt
set gloopcnt = gloopcnt - 1
end if
--put gloopcnt
updatestage
end

On AnimateSUB
puppetSprite 8, True

global gloopcntS
--if the puppet of sprite 8 = true then
if (random(1) = 1) and (gloopcntS < 1) then
  set gloopcntS = 9
end if

if gloopcntS > 0 then
  set the castNum of Sprite 8 to 198 + gloopcntS
  set gloopcntS = gloopcntS - 1
end if
updatestage
end

On Animateintroمار
puppetSprite 12, True

global gloopcnted
if (random(1) = 1) and (gloopcnted < 1) then
  set gloopcnted = 6
end if

if gloopcnted > 0 then
  set the castNum of Sprite 12 to 351 + gloopcnted
  set gloopcnted = gloopcnted - 1
end if
updatestage
end

Score Script6

on exitFrame
  buttonrolls
  level
  greyRoll
  go "menu"
end
Score Script 18

on exitFrame
  global gScreen
  puppetSprite 36, True
  puppetSprite 37, True
  put the frame into gScreen
  buttonrolls
  OrangeRoll
  DolphinRoll
  OrcaRoll
  ColorCursorOrange
  go the frame
end

Score Script 19

on exitFrame
  go "Hammerhead"
end

Score Script 20

on exitFrame
  go "Manta Ray"
end

Score Script 21

on exitFrame
  buttonroll
  go the frame
end
Score Script83

on exitFrame
delay 9 * 60
end

Score Script84

on exitFrame
  AnimateWater
  buttonrolls
  level
  GreyRoll
go "credits"
end

Score Script85

on exitFrame
  AnimateWater
  buttonrolls
  level
  GreyRoll
  puppetSprite 9, False
  if soundbusy(2) then sound stop 2
  updateStage
  go "Credits"
end

Score Script86

on exitFrame
  delay 12 * 60
end
Score Script

on exitFrame
  sound playfile 6, "Final Intro Real"
  updatestage
end

repeat while the stilldown
  set the castnum of sprite 28 to the number of cast "DictionaryInverse"
  updatestage
end repeat

Script of Cast Member

on mouseUp
  puppetSprite 36, False
  puppetSprite 37, False
end

Script of Cast Member

on mouseUp
  puppetSprite 36, False
  puppetSprite 37, False
end
Script of Cast Member125: Dictionaryroll

on mousedown
  --to swap cast
  puppetSprite 36, False
  puppetSprite 37, False
  repeat while the stilldown
    set the castNum of sprite 28 to the number of cast "Dictionaryinverse"
    updateStage
  end repeat
end

on mouseUp
  PuppetSound "LittleButtons"
  puppetSprite 39, True
  set the castNum of sprite 28 to the number of cast "Dictionaryroll"
  set the castNum of sprite 39 to the number of cast "Warning underconstructionindex"
  updateStage
  set the timer = 0
  repeat while the timer < 3 * 60
    end repeat
  set the castNum of sprite 39 to the number of cast "Dictionarywarning"
  updateStage
  puppetSprite 39, False
end
Script of Cast Member127: MMAroll

On mousedown
    repeat while the stilldown
        set the castNum of sprite 35 to 112
        updateStage
    end repeat
end

On mouseUp
--To send users to specific sections in MMA--use frame numbers
puppetSound "Click"
global gMMA
puppetSprite 36, False
puppetSprite 37, False
puppetSprite 42, False
set the castNum of sprite 35 to the number of cast "MMAroll"
updateStage
if (gMMA >= 29 and gMMA <= 56) then
    go "Dolphin" of movie "MMA"
end if
if (gMMA >= 62 and gMMA <= 83) then
    go "Flounder" of movie "MMA"
end if
if (gMMA >= 89 and gMMA <= 110) then
    go "Killerwhale" of movie "MMA"
end if
if (gMMA >= 115 and gMMA <= 147) then
    go "Turtle" of movie "MMA"
end if
if (gMMA >= 153 and gMMA <= 194) then
    go "Hammerhead" of movie "MMA"
end if
if (gMMA >= 200 and gMMA <= 220) then
    go "Tuna" of movie "MMA"
end if
if (gMMA >= 226 and gMMA <= 264) then
    go "Mantaray" of movie "MMA"
end if
if (gMMA >= 268 and gMMA <= 306) then
    go "SeaSnake" of movie "MMA"
end if
if (gMMA >= 1 and gMMA <= 28) then
    go gMMA of movie "MMA"
end if
if gMMA >= 307 then
    go gMMA of movie "MMA"
end if
end
On mousedown
  puppetSprite 36, False
  puppetSprite 37, False
  repeat while the stilldown
    set the castNum of sprite 29 to the number of cast "Helpinverse"
    set the castNum of sprite 43 to the number of cast "blacklevel"
    updateStage
  end repeat
end

on mouseUp
  Global gHelp
  puppetSound "Click"
  set the castNum of sprite 29 to the number of cast "Helproll"
  put the frame into gHelp
  put gHelp
  puppetSprite 4, False
  puppetSprite 5, False
  puppetSprite 6, False
  puppetSprite 7, False
  puppetSprite 8, False
  puppetSprite 9, False
  puppetSprite 10, False
  puppetSprite 11, False
  puppetSprite 12, False
  puppetSprite 13, False
  puppetSprite 14, False
  puppetSprite 15, False
  puppetSprite 16, False
  puppetSprite 43, False
  updateStage
  puppetTransition 23
  go "Help"
end
On mousedown
  puppetSprite 36, False
  puppetSprite 37, False
  repeat while the stillDown
    set the castNum of sprite 29 to the number of cast "Helpinverse"
    set the castNum of sprite 43 to the number of cast "blacklevel"
    updateStage
  end repeat
end

on mouseUp
  Global gHelp
  puppetSound "click"
  set the castNum of sprite 29 to the number of cast "HelprollHelp"
  puppetSprite 4, False
  puppetSprite 5, False
  puppetSprite 6, False
  puppetSprite 7, False
  puppetSprite 8, False
  puppetSprite 9, False
  puppetSprite 10, False
  puppetSprite 11, False
  puppetSprite 12, False
  puppetSprite 13, False
  puppetSprite 14, False
  puppetSprite 15, False
  puppetSprite 16, False
  puppetSprite 43, False
  updateStage
  puppetTransition 23
  go gHelp
end
Pretestinfo

Script of Cast Member 143: Helpmovie

on mouseDown
  puppetSprite 36, False
  puppetSprite 37, False
  updateStage
end

on mouseUp
  puppetSound "Mouse Down"
  updateStage
end

Script of Cast Member 144: Tunaicon

on mouseUp
  puppetSound "Mouse Down"
  puppetSprite 36, False
  puppetSprite 37, False
  updateStage
  go "Tunamovie"
end

Script of Cast Member 145: Hammer Icon

on mouseUp
  puppetSound "Mouse Down"
  puppetSprite 36, False
  puppetSprite 37, False
  updateStage
  go "Hammermovie"
end

Script of Cast Member 149: Mania Icon

on mouseUp
  puppetSound "Mouse Down"
  puppetSprite 36, False
  puppetSprite 37, False
  updateStage
  go "Maniamovie"
end
Script of Cast Member 146: Turtle Icon

on mouseUp
  puppetSound "Mouse Down"
puppetSprite 36, False
puppetSprite 37, False
updateStage
  go "Turtlemovie"
end

Script of Cast Member 147: Snake Icon

on mouseUp
  puppetSound "Mouse Down"
puppetSprite 36, False
puppetSprite 37, False
updateStage
  go "SnakeMovie"
end

Script of Cast Member 148: Whale Icon

on mouseUp
  puppetSound "Mouse Down"
puppetSprite 36, False
puppetSprite 37, False
updateStage
  go "WhaleMovie"
end

Script of Cast Member 149: Manta Icon

on MouseUp
  puppetSound "Mouse Down"
puppetSprite 36, False
puppetSprite 37, False
updateStage
  go "MantaMovie"
end
Script of Cast Member 150: Floundericon

on mouseUp
  puppetSound "Mouse Down"
  puppetSprite 36, False
  puppetSprite 37, False
  updateStage
  go "Floundermovie"
end

Script of Cast Member 151: Dolphinicon

on mouseUp
  puppetSound "Mouse Down"
  puppetSprite 13, False
  puppetSprite 14, False
  updateStage
  go "Dolphinmovie"
end
on mouseUp
global gMap
   puppetSound "CluckMap"
   set the castNum of sprite 16 to the number of cast "Mapvert"
   updateStage
   puppetTransition 23
   puppetSprite 4, False
   puppetSprite 5, False
   puppetSprite 6, False
   puppetSprite 7, False
   puppetSprite 8, False
   puppetSprite 9, False
   puppetSprite 10, False
   puppetSprite 11, False
   puppetSprite 12, False
   puppetSprite 13, False
   puppetSprite 14, False
   puppetSprite 15, False
   puppetSprite 16, False
   puppetSprite 43, False
   go gMap
end

on mouseDown
   puppetSprite 46, True
   repeat while the stilldown
      set the castNum of sprite 46 to the number of cast "Stararrow"
   end repeat
end

on mouseUp
   puppetSound "click"
   set the castNum of sprite 46 to the number of cast "Stararrow"
   updateStage
   puppetTransition 51
   go marker (+1)
   puppetSprite 46, False
   updateStage
end
Script of Cast Member216: Larrow

On mouseDown
  puppetSprite 36, False
  puppetSprite 37, False
  puppetSprite 47, True
  repeat while the stilldown
    set the castNum of sprite 47 to the number of cast "Ltinversearrow"
    updateStage
  end repeat
end

on mouseUp
  puppetSound "click"
  set the castNum of sprite 47 to the number of cast "Ltarrown" updateStage
  puppetTransition 51
  go marker (-1)
  puppetSprite 47, False
  updateStage
end

Script of Cast Member220: Rarrow

On mouseDown
  puppetSprite 46, True
  repeat while the stilldown
    set the castNum of sprite 46 to the number of cast "Rtinversearrow"
    updateStage
  end repeat
end

on mouseUp
  puppetSound "click"
  set the castNum of sprite 46 to the number of cast "Rtarrown" updateStage
  puppetTransition 51
  go marker (+1)
  puppetSprite 46, False
  updateStage
end
Script of Cast Member 224: Rarrow

On mouseDown
  puppetSprite 46, True
  repeat while the stilldown
    set the castNum of sprite 46 to the number of cast "RtInversearrow"
    updateStage
  end repeat
end

on mouseUp
  puppetSound "click"
  set the castNum of sprite 46 to the number of cast "Rarrow"
  updateStage
  puppetTransition 51
  go marker (+2)
  puppetSprite 46, False
  updateStage
end

Script of Cast Member 225: Larrow

On mouseDown
  puppetSprite 47, True
  repeat while the stilldown
    set the castNum of sprite 47 to the number of cast "LtInversearrow"
    updateStage
  end repeat
end

on mouseUp
  puppetSound "click"
  set the castNum of sprite 47 to the number of cast "Larrow"
  updateStage
  puppetTransition 51
  go marker (-1)
  puppetSprite 47, False
  updateStage
end