# Project Electus Plan

*Michael Peterson, Jack Bauman, Michael Feng, Alexis Lopez*

# *Table of Contents*

# *I. Scope Management*

In *Project Electus* the player takes the role of a super hero born in a laboratory, who finds himself in a world of evil. The game will be a 2-D side scrolling, 8-bit, action-adventure platformer, the player will advance through the levels fighting different enemies using basic combat skills such as dodge, punch, and jump, and plethora of player selected attributes. These attributes will be obtained by gaining experience from completing levels and defeating enemies. Attributes will vary in power and type depending on how the player chooses to improve them. Levels will contain many obstacles alongside unique enemies to impede the player from reaching his goal, which is defeating the final villain of each level.

# *II. Requirements Management*

# Function

In Project Electus the player takes the role of a super hero born in a laboratory, who finds himself in a world of evil. The game will be a 2-D side scrolling, 8-bit, action-adventure platformer. The player will advance through the levels fighting different enemies using basic combat skills such as dodge, punch, and jump, and a plethora of player selected attributes. These attributes will be obtained by gaining experience from completing levels and defeating enemies. Attributes will vary in power and type depending on how the player chooses to improve them. Levels will contain many obstacles with unique enemies to impede the player from reaching his goal, which is defeating the final villain of each level.

The game will include several robust and unique features, such as customizable characters, player chosen attributes, and varying and challenging levels. In the beginning of the game, the player will select a starting power and will choose to upgrade his abilities to create the player's customizable hero. As the player progresses throughout the game he will encounter unique and specialized enemies and levels. In order to win the game the player must beat the level and the final boss.

Project Electus will have three in-game menus that consists of a main menu, a pause menu, and a character menu. There will be multiple menu options available depending on the menu the player is using. When a player starts the game the main menu will appear, it will have new game, load game, and exit. When a player pauses the game it calls upon the pause menu

which can save game, return to main menu, and go to the player menu. When the player menu is selected it will show the hero's attributes and allow the player to improve his powers.

Enemies in Project Electus consist of evil mutant, toxic blobs and red-eyed robots. When a player gets close to an enemy, it will move toward the user and try to hurt the player. Some of the enemy's many features consist of melee attacks and far-ranged attacks based on the enemy. Their main objective of the enemies is to try to lower your health to zero.

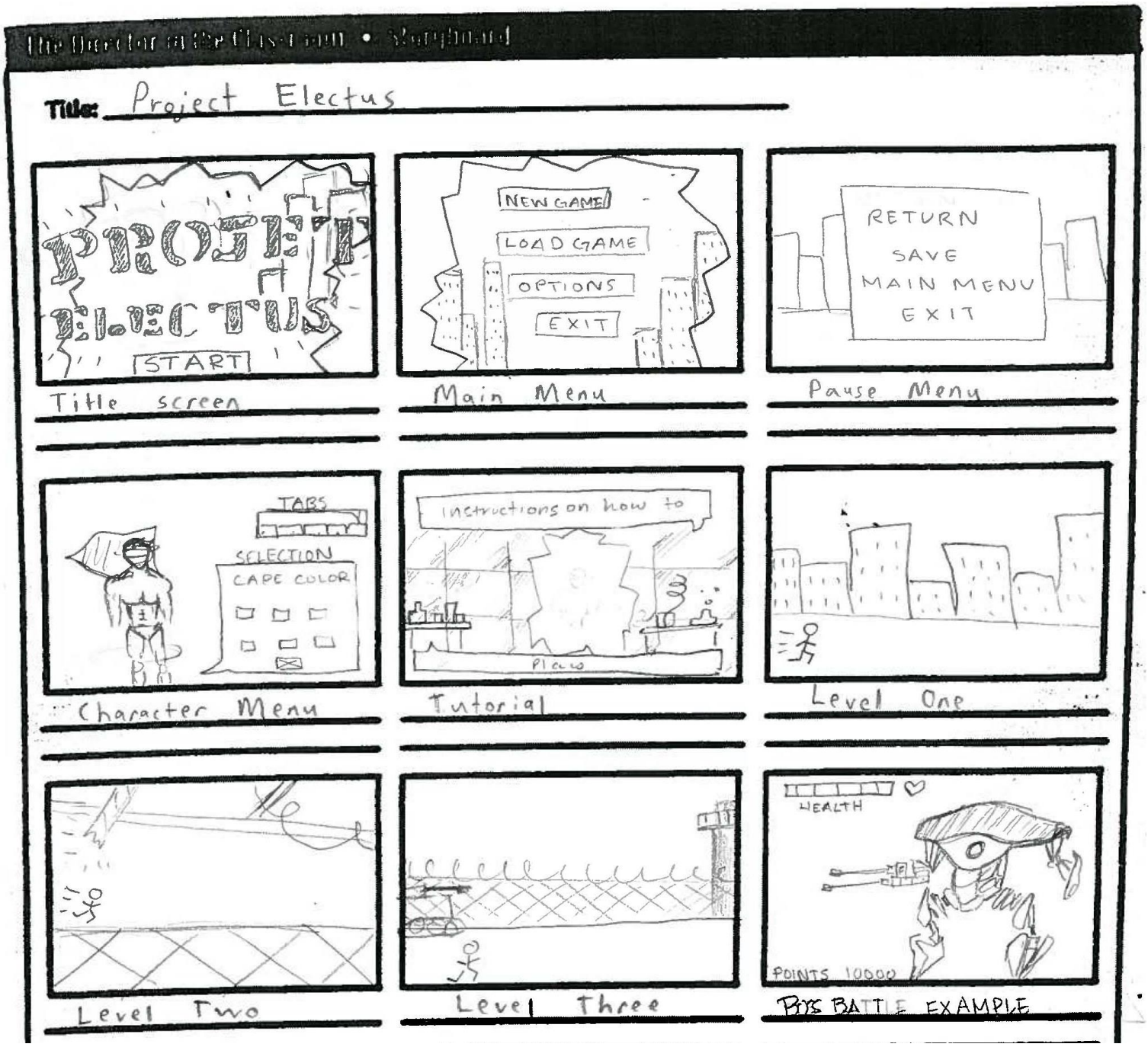
As the player progresses throughout the levels, the superhero will defeat enemies and gain experience, by completing a level and destroying enemies. With this experience the player may choose to improve his powers to the extent of how much experience he gained. In the game the hero can only level up, but cannot level down.

As the player progresses through the game experience will be given to the player as he wins levels and destroys enemies. The enemies will drop experience, which look like small orbs, from them after they are destroyed. Each enemy will yield a certain amount of experience and the completion of each level will grant the player experience. The player can then use his experience points to improve his current skill set.

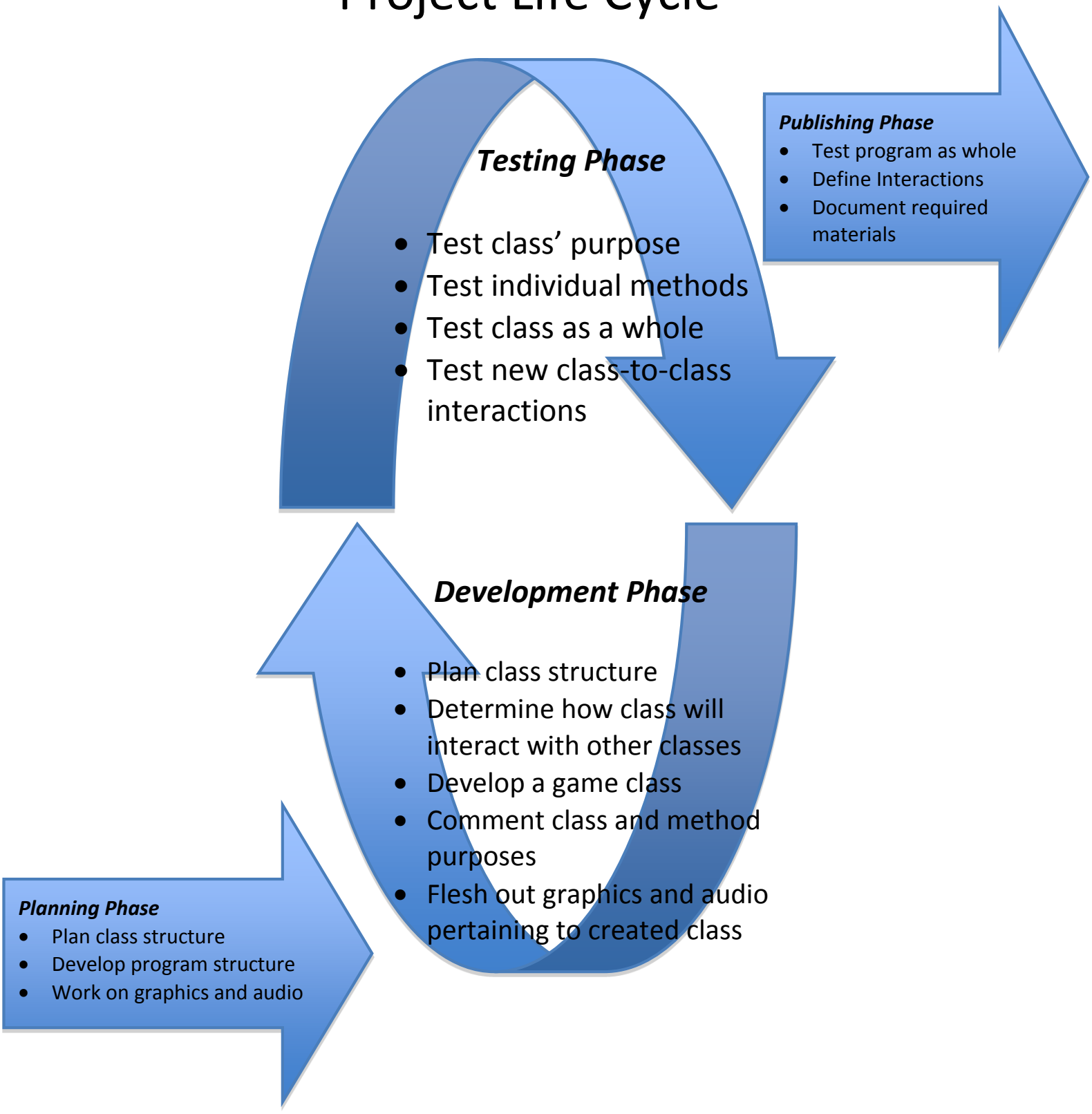
The window's view will center on the player controlled character. The view of the window will move across a larger image to reveal the individual platforms, items, and enemies of each level. The player will move across the screen walking and jumping but will also be able to fly and teleport if those abilities are unlocked.

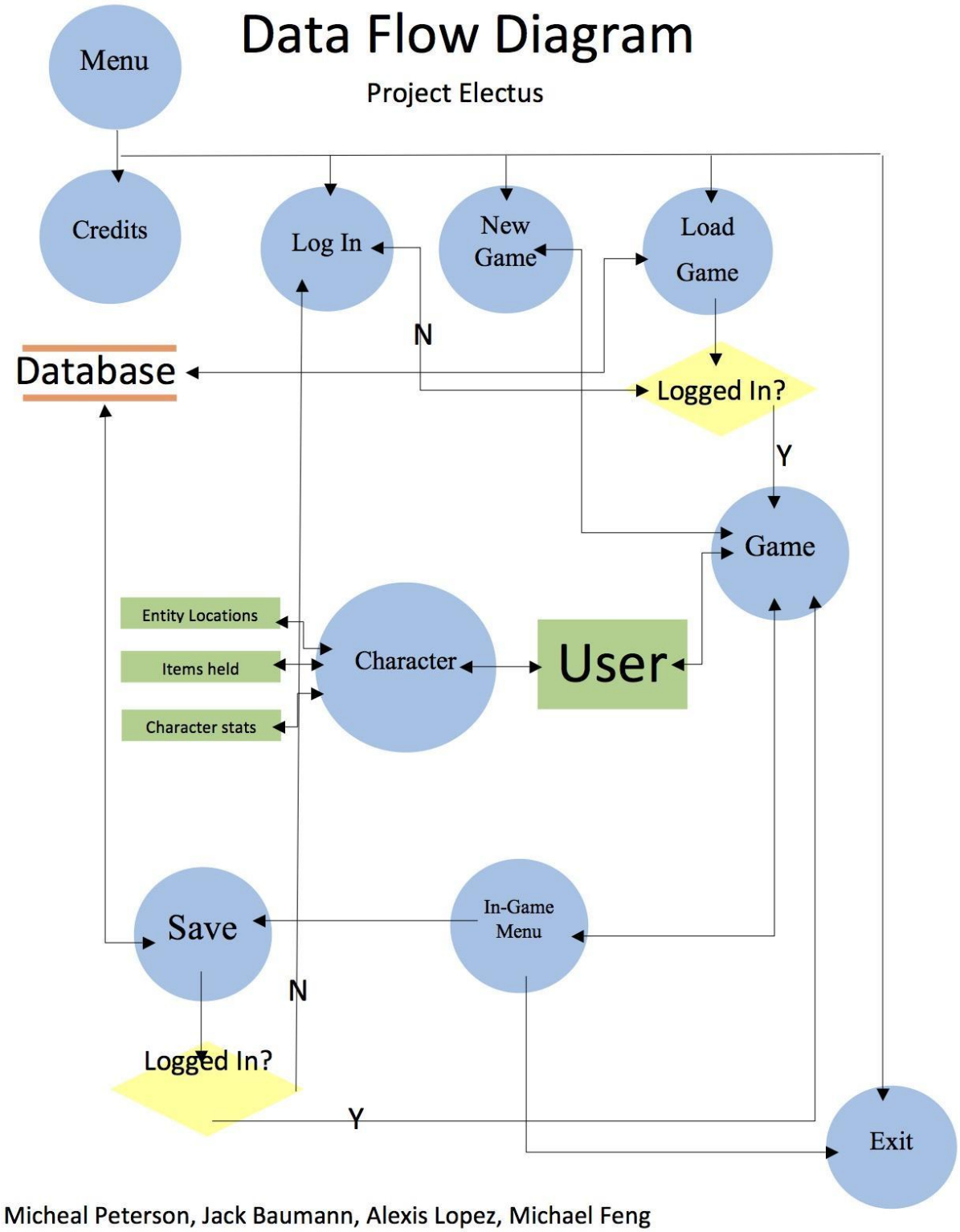
In the game, Project Electus, the player designs their own custom hero by selecting the hero's powers and attributes. The hero must find a way through a world of evil while simultaneously leveling up his powers and getting stronger. The hero will get stronger with upgrades acquired with experience points that he collects from enemies, such as robots and mutant blobs, and by completing levels.

# Storyboard

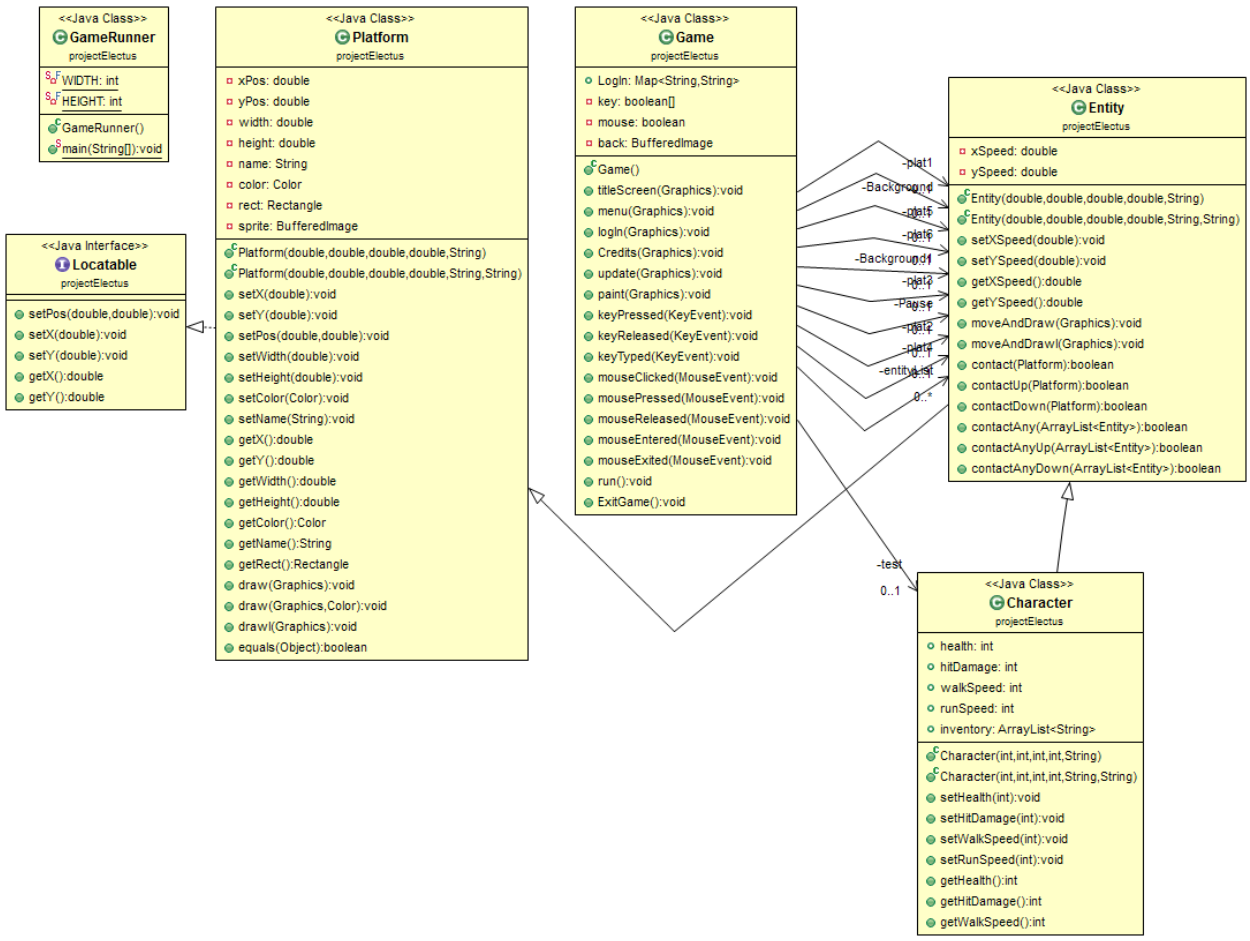

Title: Project Electus
PROJECT ELECTUS
START
Title screen
NEW GAME
LOAD GAME
OPTIONS
EXIT
Main Menu
RETURN
SAVE
MAIN MENU
EXIT
Pause Menu
TABS
SELECTION
CAPE COLOR
Character Menu
Instructions on how to
Tutorial
Level One
Level Two
Level Three
HEALTH
POINTS 10000
BOSS BATTLE EXAMPLE

# Project Life Cycle






Micheal Peterson, Jack Baumann, Alexis Lopez, Michael Feng


<<Java Class>>
GameRunner
projectElectus
WIDTH: int
HEIGHT: int
GameRunner()
main(String[]):void
<<Java Interface>>
Locatable
projectElectus
setPos(double,double):void
setX(double):void
setY(double):void
getX():double
getY():double
<<Java Class>>
Platform
projectElectus
xPos: double
yPos: double
width: double
height: double
name: String
color: Color
rect: Rectangle
sprite: BufferedImage
Platform(double,double,double,double,String)
Platform(double,double,double,double,String,String)
setX(double):void
setY(double):void
setPos(double,double):void
setWidth(double):void
setHeight(double):void
setColor(Color):void
setName(String):void
getX():double
getY():double
getWidth():double
getHeight():double
getColor():Color
getName():String
getRect():Rectangle
draw(Graphics):void
draw(Graphics,Color):void
drawI(Graphics):void
equals(Object):boolean
<<Java Class>>
Game
projectElectus
LogIn: Map<String,String>
key: boolean[]
mouse: boolean
back: BufferedImage
Game()
titleScreen(Graphics):void
menu(Graphics):void
logIn(Graphics):void
Credits(Graphics):void
update(Graphics):void
paint(Graphics):void
keyPressed(KeyEvent):void
keyReleased(KeyEvent):void
keyTyped(KeyEvent):void
mouseClicked(MouseEvent):void
mousePressed(MouseEvent):void
mouseReleased(MouseEvent):void
mouseEntered(MouseEvent):void
mouseExited(MouseEvent):void
run():void
ExitGame():void
-plat1
-Background
-plat5
-plat6
-Background1
-plat3
-Pause
-plat2
-plat4
-entityList
0..*
<<Java Class>>
Entity
projectElectus
xSpeed: double
ySpeed: double
Entity(double,double,double,double,String)
Entity(double,double,double,double,String,String)
setXSpeed(double):void
setYSpeed(double):void
getXSpeed():double
getYSpeed():double
moveAndDraw(Graphics):void
moveAndDrawI(Graphics):void
contact(Platform):boolean
contactUp(Platform):boolean
contactDown(Platform):boolean
contactAny(ArrayList<Entity>):boolean
contactAnyUp(ArrayList<Entity>):boolean
contactAnyDown(ArrayList<Entity>):boolean
-test
0..1
<<Java Class>>
Character
projectElectus
health: int
hitDamage: int
walkSpeed: int
runSpeed: int
inventory: ArrayList<String>
Character(int,int,int,int,String)
Character(int,int,int,int,String,String)
setHealth(int):void
setHitDamage(int):void
setWalkSpeed(int):void
setRunSpeed(int):void
getHealth():int
getHitDamage():int
getWalkSpeed():int

# Instructions

## Moving

1. The player cam move his hero by using the W, A, S, and D buttons.
2. By pressing A and D the player can move left and right.
3. Pressing S makes the hero crouch and pressing W makes the hero jump (double-tapping W allows the hero to fly, if he has that ability).

## Using your powers

1. The numbers 1-9 on top of the keyboard are slots that that can hold powers for the hero to use.
2. By pressing a number 1-9 it activates the power that corresponds to it.
3. In order to use the activated power press the space bar.
4. The numbers 1-9 may or may not have a corresponding power if you don't have nine powers or if you do not assign the number a power to hold.

## Combat

1. Enemies, such as robots, will try to attack your hero throughout the game.
2. If you are hit by one of their attacks you will lose health. If all your health is lost you die.
3. If an enemy attacks you, try to dodge the attack by moving away from it.
4. In order to defeat the robot you must use hit the robot with your powers by pressing spacebar.
5. Throughout the game you should try to dodge the enemies' attacks and destroy the enemies with your attacks.

# Tutorial

## The Basics of Moving

1. The character can move around the map using the W, A, S, and D buttons.
2. The “A” button moves the player left and the “D” button moves the player right.
3. When the “S” button is pressed your character will crouch.
4. The “W” button allows the player to jump. Double tapping “W” also allows the player to fly if he has that ability.

## Selecting Powers and Using Them

1. Each players’ powers will be stored in the numbers 1 through 9 on top of the keyboard.
2. Each number will correlate to one of the players’ selected powers.
3. When a number is pressed, the power that correlates to that number is activated.
4. To use this activated power press the space bar.

## Attacking the Enemy

1. When enemies, such as robots, try to attack the player via lasers or melee attack the player should try to dodge it by moving away from the attack.
2. If you are hit however your health will go down and once all your health is gone you will lose the game.
3. Using the variety of powers, players will attack the enemy and try lower their health to zero and destroy them.
4. The large selection of powers allows many techniques and strategies to be used to defeat the enemies throughout the game. So create a hero develop a strategy using his powers and good luck.

# *III. Schedule Management*

# Jobs and Descriptions

Michael Peterson: Lead Programmer and will create most of the code. He is also the Project manager and will make sure everyone is on track.

Jack Baumann: Assistant programmer who will help create the code along with the lead programmer. He will also help the lead programmer debug any errors found in the code.

Alexis Lopez: Creates the artwork, graphics, and sounds for the game. So all images and sounds for the game will be designed by her.

Michael Feng: Writes documentation for the game and assist in programming. He will fill out the forms and documents required to participate in events.

# Timeline Table

## Project Timeline, Printed Documents

Filename: ISinCS Timeline Printed Material (nonCode).doc

| Draft Completed | Date Finalized | Task | Person Responsible |
|---|---|---|---|
| 09/27 | 09/27 | **Team and Project Software Name** | All |
| 09/27 | 09/27 | **Job Descriptions** | All |
| 09/30 | 10/02 | **Function** | Michael Feng, Jack Baumann |
| 10/01 | 10/03 | **Project Plan (what, why, who, when, and more…)** | Micheal Peterson |
| 10/10 | 10/11 | **System Development Life-Cycle** | Jack Baumann |
| 10/13 | 10/14 | **Storyboard of major events** | Alexis Lopez |
| 10/16 | 10/16 | **Title Page** | Alexis Lopez |
| 10/27 | 10/29 | **Data Flow Diagram** | Micheal Peterson |
| 11/17 | 11/19 | **UML Class/Methods Diagram complete** | Micheal Peterson |
| 11/18 | 11/19 | **Hardware and Software Requirements Specifications** | Jack Baumann |
| 11/17 | 11/19 | **User Manual/Tutorial basic outline complete** | Michael Feng |
| 11/19 | 11/21 | **Functional Requirements Page(s)** | Jack Baumann |
| 11/30 | 11/30 | **Table of Contents Manager** | Michael Feng |
| 12/17 | 12/20 | **Updated Data Flow, Physical Flow, UML diagrams and Storyboard** | Micheal Peterson |
| 12/18 | 12/20 | **Presentation PowerPoint draft complete** | Alexis Lopez |
| 01/05 | 01/07 | **Testing chart complete for all of the project** | Jack Baumann |
| 01/06 | 01/10 | **All external documentation complete** | Michael Feng |
| 01/12 | 01/14 | **All presentation material complete including 'props'** | Alexis Lopez |
| 01/15 | 01/15 | **1st Presentation read-through** | All |
| 01/17 | 01/21 | **Readme.txt complete** | Michael Feng |
| 01/24 | 01/25 | **Disk created for competition** | Micheal Peterson |
| 01/30 | 01/30 | **Presentation timed** | All |
| 02/03 | 02/03 | **2nd Presentation practice** | All |
| 02/10 | 02/10 | **Presentation finalized** | All |
| 02/12 | 02/12 | **3rd Presentation practice with audience** | All |
| 02/20 | 02/20 | **4th Presentation practice w/mock judges** | All |
| 02/25 | 02/25 | **Teacher has all completed material (notebook and cd)** | All |
| 03/04 | 03/04 | **Mock presentation using Skype** | All |
| 4/30 | 4/30 | **Actual presentation with judges** | All |

# Timeline

By mid-November, the data-flow diagram, table of contents, physical flow diagram, UML class diagram, functional requirement page(s), development life cycle, and software requirement specifications will be complete. After all of this is completed, programming will occur. These include manager class with 'stub' worker classes (complete with comments and JavaDocs). By December 15th, one-quarter of the project will be completed and testing chart will be done for the first quarter as well. In addition, the user manual will be updated complete with the basic outline. By December 20th another quarter of the project will be coded for, and JavaDocs and comments will be finished for that section. Updated data flow, physical flow, UML diagrams and storyboard will be completed by this time as well. By January 5th, the third quarter will be completely coded for and commented. The testing chart will be done as well.

# *IV. Quality Management*

Throughout the project, we will be sure to comment above every method and class describing the purpose, projected outcome, and any complicated algorithm or logic. Each part of the project (I.E. classes, methods, interfaces, etc.) will be mentioned in the “Files” category in the comments before the import statements and class header. Classes will also be cross checked by other team members to ensure validity and proper logic, which will allow us to further collaborate on ideas, increase efficiency, and solve errors with ease. Additionally, when an error occurs, we will add a block in the code where the code will generate everything before it, to help troubleshoot minor errors. As for major errors, they will be recorded and fixed through various testing and team collaboration. The testing we will perform

# *V. Resource Management*

Our team will use NetBeans for our code in Project Electus, so every team member will be required to have a personal computer with NetBeans or access to a school computer with NetBeans. Our information will be mainly shared through email and GoogleDocs, so that everyone will have easy access to the information and it will be easy to edit. The artwork will be done using Adobe Illustrator and Adobe Photoshop, while all the code is written in java. All computers used for the project will need to meet these software requirements to work effectively on Project Electus.

# *VI. Communications Management*

For communications through the team, we will use Google Docs to easily share needed files. These files can then be downloaded to home or school computers, and effectively altered, and uploaded and updated to the online driver. Additionally, we will use email to communicate ideas and deadlines, and other necessary information we do not convey face to face. Also, each member has a list of phone numbers and emails to communicate deadlines and refer them to the Google Docs shared folder. For each file, we are able to log when each user uploaded it and altered it, which will help us determine who did tasks correctly, and if they did not perform the tasks correctly, communicate with them through email or text message to advise them what to alter and correct. There are subfolders inside the Project Electus folder to categorize and organize needed files, which include data diagrams, plans, summaries, class files shared by team members, graphics, audio, and outlines for further tasks to be completed. This ease of access allows all the team members to effectively communicate and share files without chaos or loss of time. These forms of communication will also allow the team members to regulate each other's performance and aid in the overall project time management.

# *VII. Risk Management*

Of the many risks we face as a team the one main risk is the amount of time it takes to complete things. After all if we can't finish our project in time then we can't compete. So in order to turn things in on time we will try to follow the timeline that we have so everything is orderly and on time. Our next big risk we must face is finding days in which the team can get together to cooperate. However, this problem can be fixed because we can communicate easily over the internet to stay connected. We can also share work over the internet if we can't meet in person.

# *VIII. Procurement Management*

Over the course of this project our team will work individually on parts of the project and share his/her work with the rest of the team. If someone falls behind it will be the project manager's job to get him back on track. Any work that can't be completed at school will then be done from home so that we don't fall too far behind. Every day we will collaborate together by sharing our work over the internet. If we need to get something done in a short period of time then the project manager will choose a day to meet after school and we will all work with each other in a group. Using these methods we should be able to finish our project on time and manage our time well.