A retro space combat game by Chad Fillion

Designed and developed as a throwback to the classic 80's arcade games, Space Attack launches players into a galaxy of Alien enemies in an endurance race to attain the highest score possible.

This process book reviews early project ideas, late stage development screen shots as well as the finished "game manual" for users to learn how to play the game.

PLAYER SHIP | MISSILE POWER-UP | STANDARD MISSILE | MISSILE POWER-UP | QUAD MISSILE

Player Score: 7100 | High Score: 7100
Level 7

Game Summary

As the pilot of the defending ship your mission is to destroy as many of the oncoming enemies as possible without hitting them as they swarm at you.

Players are challenged with dodging attacks from above, as well as their flank. Special weapon power-ups help players by upgrading their ships missiles from single shot to double shot, and from double shot to quadruple shot. But beware...shoot a power-up and you will loose all of your upgrades.

Space Attack

Movement
Fire Missile
Space Bar

- 5 - 25 points
- 15 - 75 points
- 50 - 125 points
- 100 - 200 points
??? - 150 - 500 points
??? - 500+ points

Missile Upgrades:

Catch Power-Ups to apply missile upgrades to your ship.

Shooting upgrades will remove all upgrades you have gained

First upgrade   Second upgrade

Back

This Game summary was developed as a final entry into the discussion board when presenting the idea to the class. After this post was made, the idea for developing a game manual came to mind. The thought was to design something akin to the classic Atari and Nintendo manuals I had as a child.

"Alien Armada"

As an elite celestial defender of your home planet you board your spaceship and head into orbit to help fend off the multiple waves of alien attackers. Similar to the "Space invaders" of the heyday of arcade games, this game is solely based on score. Players will have a set number of lives and will fight waves of aliens in their formation until the player's lives are exhausted. Examples from the web are shown here as concept art. Original artwork will be created.

As the development of this game will be an intense process I have broken up the ultimate goals of development into 3 tiers.

1st tier:
Player has 3 lives and a new life is earned at certain points
Player starts with a ship and faces 4 types aliens.
Aliens appear as an army "block" and shift back and forth.
Player will fire a weapon and weapon will kill enemies.
Enemies will fire back at the player.
level phases for "completion" of waves of attacks.
Various scores will be given for the different aliens.

2nd tier:
Barricades separate enemies from player and break down with hits
Force field power up / protection shield

3rd tier:
Swarm aliens introduced which bombard player
Different bonus weapon added to player
High score system

Image example of a "Player" ship

http://dannycruz.com/wp-content/uploads/2009/12/
dannycruz_design_ravx_spaceship.png

Image examples of a "Alien" ships

http://3.bp.blogspot.com/-r-IvarVka-E/UL-B8xm0FtI/
AAAAAAAAAOU/51z6j9-Xfwc/s1600/5concepts.png
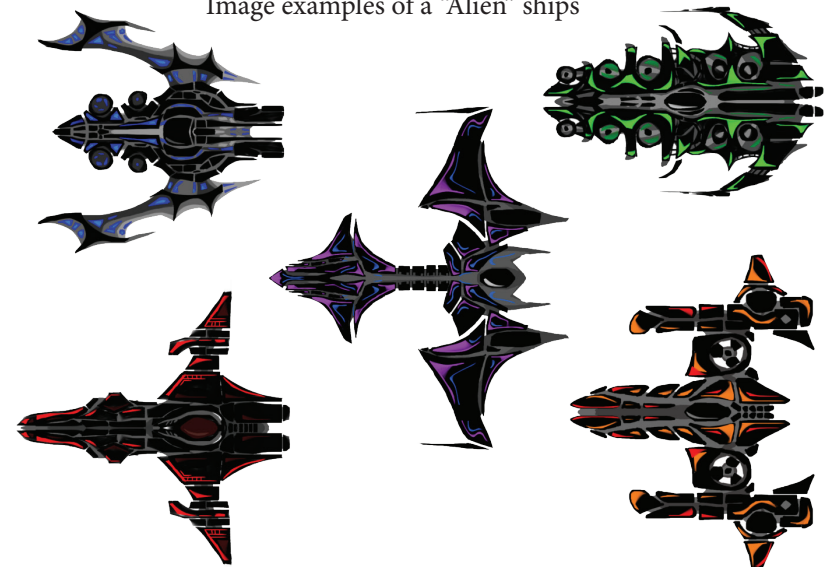
Early in the development process I knew I wanted this to be a score driven game with little "action" outside of the simple shooter mechanics. Concept art depicted a series of alien ships the player would face off against. You can see here the tiers of milestones designed to set the project pace. Some of these elements never made it into the game.

Scope outline:

Sprite sheet needs:

Player ship	Alien/Enemy ship (4-6 versions)
Player Bullet/missile	Alien bullet/missile
Damaged player ship	damaged enemy ship (4-6 versions)
Explosion clip	Title screen
Menu screen	Help screen
Game over screen	Background space scenes

If time permits:

Force field	Barrier/barricade (4-6 phases)
Powerup bonus item	

Programming requirements:

Help screen   - Info on game play / directions
Title/Menu screen - Main credits/nav to help & game start
Game screen - UI for game play

Starting score of 0
Starting player count (3)
Player movement within bounds (Bfly rectangle project)
Ship movement with arrows or WASD
Enemy array on screen information
enemy position and motion in relation to screen
Enemy position and approach toward player
Projectile fire with spacebar
Enemy projectile fire rate

Conditionals:

Detect if player's missile hits enemy
    If hits - play damaged ship or explosion clip
    calculate score and add appropriate points

Detect if enemy missile hits player
    If hits - play damaged ship or explosion clip

Detection of lives remaining
    If lives remain, reduce by one and begin again
      if Lives are all gone, show game over screen

Detect if missiles (player or enemy) hit barricade
    if hits barricade, change phase to reduced look
      If barricade is depleted, remove sprite

Detect if player collects power up
    if player collects power up, display force field
      allow for force field for certain time limit
      kill force field after certain time
      OR
    Detect if missiles (player or enemy) hit force field
      If hits force field, reduce capacity
        if capacity is depleted, destroy force field

Development of a scope outline helped me to determine what should be included, what may be able to be added if time permitted, the functional components that needed to be developed such as what screens would need to be created, and the pseudo code and logic for the conditional for the collision detection.

High Score

Player 1   000000   1000000

Adobe Flash Player 14

File   View   Control   Help
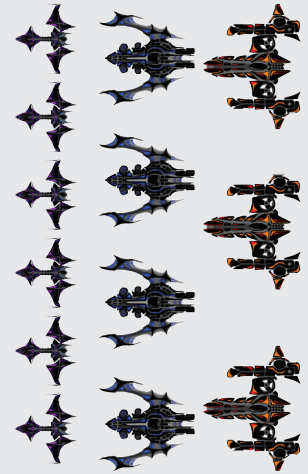
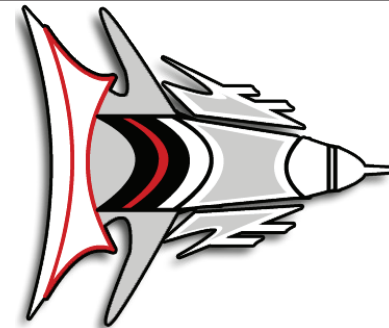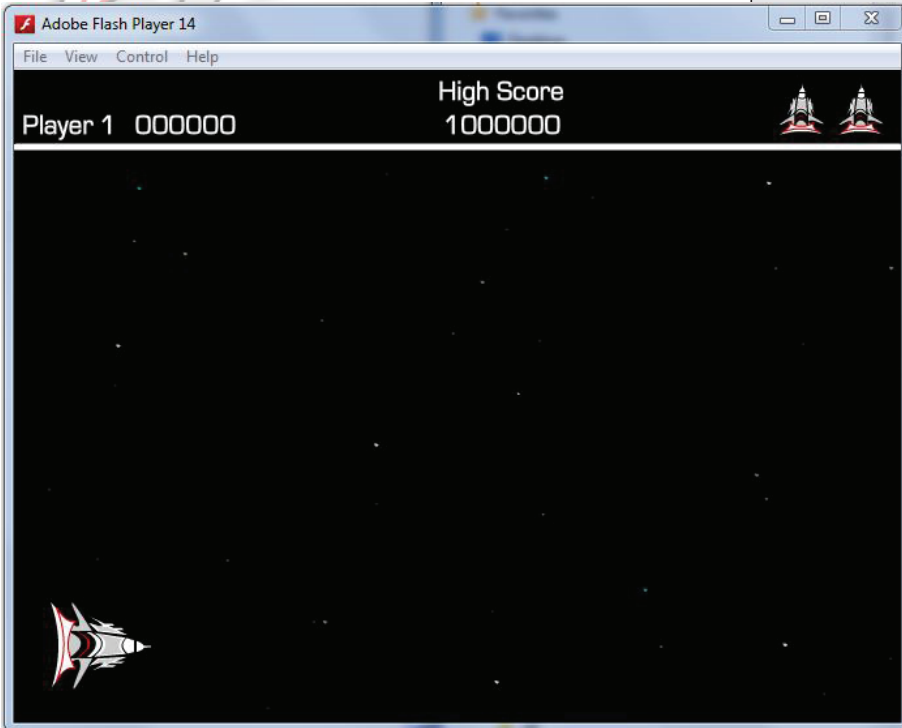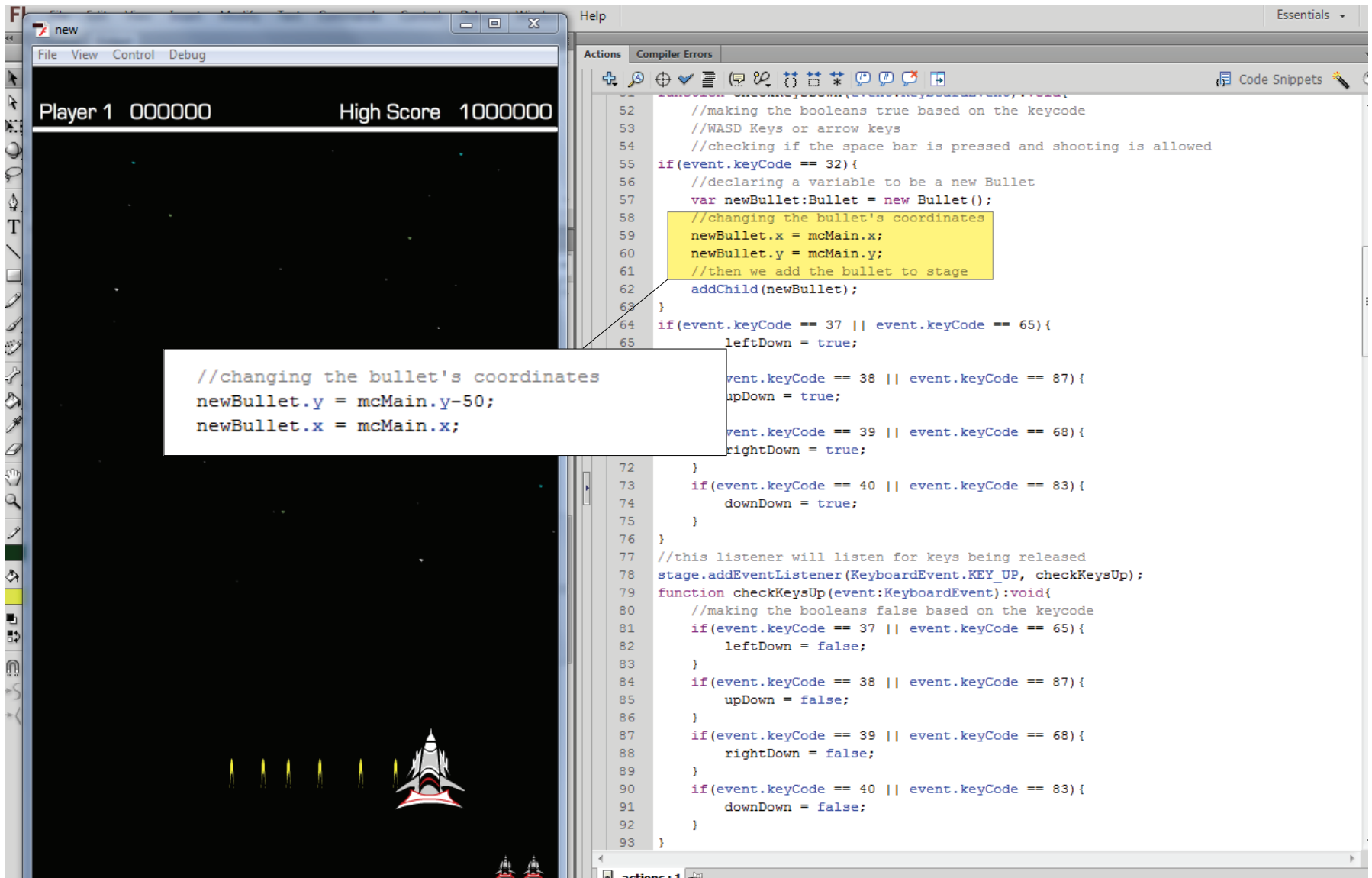MAIN MENU   HELP MENU   EXIT GAME

Early tests within the game design had the layout set up as horizontal shooter "driving" the ship from the left side of the screen towards the enemies on the right. This page shows an early layout concept, as well as an early screen grab of this format.

```
new                                    [_][□][X]     Help

File  View  Control  Debug             Actions  Compiler Errors

                                       ⊕ ⊘ ✓ ≡ ⊕ ⅋ ‡ ‡ * ⊕ ⊕ ⊕ ⊞        Code Snippets  ✦

Player 1  000000    High Score  1000000
                                       51   function checkKeysDown(event:KeyboardEvent):void{
                                       52       //making the booleans true based on the keycode
                                       53       //WASD Keys or arrow keys
                                       54       //checking if the space bar is pressed and shooting is allowed
                                       55   if(event.keyCode == 32){
                                       56       //declaring a variable to be a new Bullet
                                       57       var newBullet:Bullet = new Bullet();
                                       58       //changing the bullet's coordinates
                                       59       newBullet.x = mcMain.x;
                                       60       newBullet.y = mcMain.y;
                                       61       //then we add the bullet to stage
                                       62       addChild(newBullet);
                                       63   }
                                       64   if(event.keyCode == 37 || event.keyCode == 65){
                                       65       leftDown = true;
       //changing the bullet's coordinates
       newBullet.y = mcMain.y-50;         event.keyCode == 38 || event.keyCode == 87){
       newBullet.x = mcMain.x;            upDown = true;

                                          event.keyCode == 39 || event.keyCode == 68){
                                          rightDown = true;

                                       72       }
                                       73       if(event.keyCode == 40 || event.keyCode == 83){
                                       74           downDown = true;
                                       75       }
                                       76   }
                                       77   //this listener will listen for keys being released
                                       78   stage.addEventListener(KeyboardEvent.KEY_UP, checkKeysUp);
                                       79   function checkKeysUp(event:KeyboardEvent):void{
                                       80       //making the booleans false based on the keycode
                                       81       if(event.keyCode == 37 || event.keyCode == 65){
                                       82           leftDown = false;
                                       83       }
                                       84       if(event.keyCode == 38 || event.keyCode == 87){
                                       85           upDown = false;
                                       86       }
                                       87       if(event.keyCode == 39 || event.keyCode == 68){
                                       88           rightDown = false;
                                       89       }
                                       90       if(event.keyCode == 40 || event.keyCode == 83){
                                       91           downDown = false;
                                       92       }
                                       93   }

                                       actions : 1
```

Further testing not only revealed problems with the missile firing (*quickly solved by changing the y coordinate appropriately*) but that the horizontal layout wouldn't play as easily as a vertical. The layout of the screen was rotated 90 degrees CCW and development continued from there; including fixing other bugs in the system.

Player Score: 0   High Score: 0
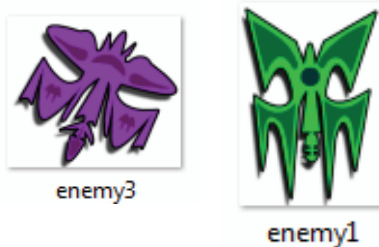
Level 1

```
139  /*******************
140  Scoring structure
141  *******************/
142  txtLevel.text = "Level 1";
143
144  if(score > 100){
145      txtLevel.text = "Level 2";
146      enemyLimit  = 30;
147      starRate = .4;
148  }
149  if(score > 200){
150      txtLevel.text = "Level 3";
151      enemyLimit = 20;
152      starRate = .6;
153      if(enemyTime2 < enemyLimit2){
154          //if time hasn't reached the limit, then just increment
155          enemyTime2 ++;
156      } else {
157          //defining a variable which will hold the new enemy
158          var newEnemy2 = new Enemy2();
159          //making the enemy offstage when it is created
160          newEnemy2.y = -1 * newEnemy2.height;
161          //making the enemy's x coordinates random
162          //the "int" function will act the same as Math.floor bu
   a bit faster
163          newEnemy2.x = int(Math.random()*(stage.stageWidth -
   newEnemy2.width));
164          //then add the enemy to stage
165          addChild(newEnemy2);
166          //and reset the enemyTime
167          enemyTime2 = 0;
168      }
169  }
170  if(score > 500){
171      txtLevel.text = "Level 4";
172      enemyLimit  = 15;
173      enemyLimit2 = 30;
174      starRate = .8;
```
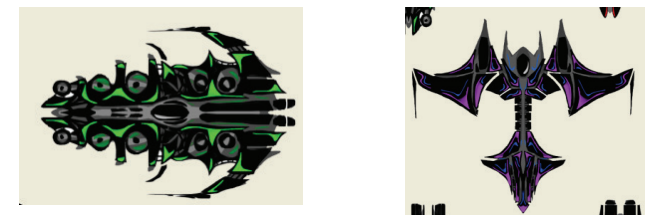
Actual game Art:

enemy3

enemy1

enemy3   missile   enemy4   ff-sheild   ff-ship   power-1

medal-s   medal-p   medal-b   extra   enemy5   enemy6
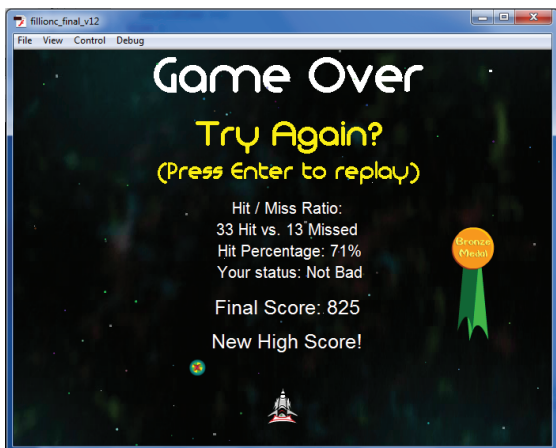
Concept Art:

Development of the various enemy ships came next. They were designed using the earlier prototype examples in mind, but with a more organic "insect" feel. In addition to the new art work, the scoring system was established within the code, setting benchmarks for higher difficulty

```
310   if(sheildTime < sheildLimit){
311       //if time hasn't reached the limit, then just increment
312       sheildTime ++;
313   } else {
314       //defining a variable which will hold the new power up
315       var newSheild = new sheild();
316       //making the power up offstage when it is created
317       newSheild.y = -1 * newSheild.height;
318       //making the power up's x coordinates random
319       //the "int" function will act the same as Math.floor but a bit faster
320   newSheild.x = int(Math.random()*(stage.stageWidth - newSheild.width));
321       //then add the power up to stage
322       addChild(newSheild);
323       //and reset the Time
324       sheildTime = 0;
325   }
326
327   //new additional life drop if player doesn't have 4 lives already
328   //(active ship is included in the count of total lives)
329       if (lives<4){
330
331       if(xtraTime < xtraLimit){
332           //if time hasn't reached the limit, then just increment
333           xtraTime ++;
334       } else {
335           //defining a variable which will hold the new additional life
336           var newExtra = new extra();
337           //making the extra life offstage when it is created
338           newExtra.y = -1 * newExtra.height;
339           //making the extra life's x coordinates random
340           //the "int" function will act the same as Math.floor but a bit faster
341           newExtra.x = int(Math.random()*(stage.stageWidth - newExtra.width));
342           //then add the extra life to stage
343           addChild(newExtra);
344           //trace('lives: '+lives);
345           //and reset the Time
346           xtraTime = 0;
347       }
348   }
```

I continued to add more functionality and opportunities for players I added in the different types of missiles for the ship, scripted in the different enemies, as well as added force fields and extra lives that are randomly generated to the screen at different times during the game. I even added ribbons to the game end screen for good performances.

Space Attack

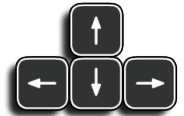A retro space combat game by Chad Fillion

How to Play -

Using the Arrow keys on the keyboard (or the WASD keys if you prefer) to control the Defender-K1 around the screen. You can move left to right as well as up and down.

The space bar will fire all missiles from the ship, regardless of which of the upgrades you have attached.

Fire Missile
Space Bar

Movement:

When the game begins you will start within a safe zone area. This safe zone is active through the first six levels of game play. If you 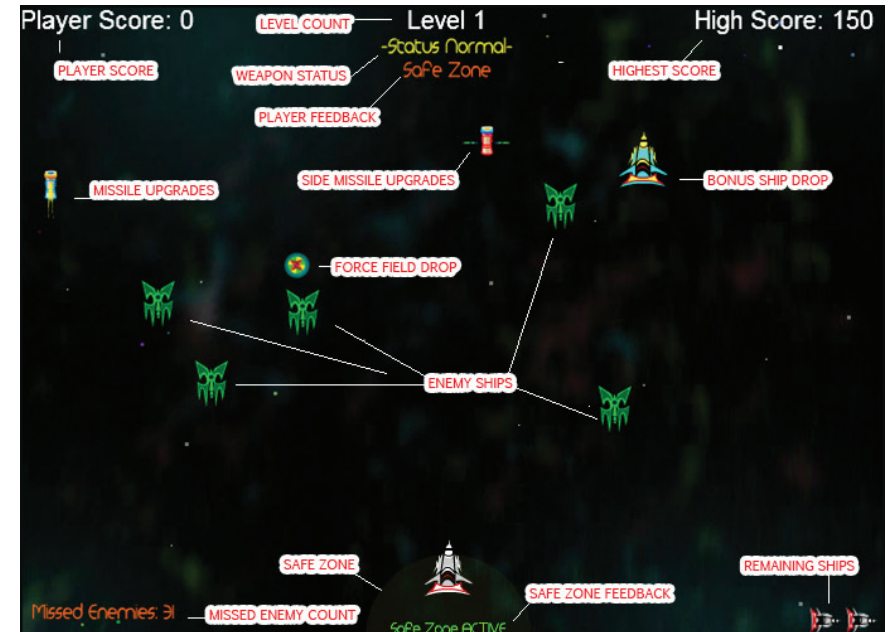lose a ship, you will regenerate in this area as well. This safe area keeps enemies from hurting you, however your missiles will not fire as often in this zone. Also, the safe area will be disabled when you gain a force field power up.

Shoot as many enemy ships as you can, catching the upgrades that drop adding more fire power to the Defender-K1. Caution though: Hitting any power ups that fall with a missile will destroy the power up and remove upgrades from the Defender-K1.


Player Score: 11825   Level 6   High Score: 11825
-Status Normal-
-Last Upgrade Dissipated-
Missed Enemies: R2
Safe Zone ACTIVE

Designed and developed as a throwback to the classic 80's arcade games, Space Attack launches players into a galaxy of Alien enemies in an endurance race to attain the highest score possible.

You are the pilot of the space ship Defender-K1. You must control the ship, and destroy as many of the oncoming enemies as possible with out colliding with them. You must use the arrow keys (or the WASD keys) to maneuver the ship side to side, and up an down on the screen to avoid collision with the swarms of enemies flying at you. Using the space bar you will fire your missiles at the enemies and try to destroy as many as possible.

Beware as the attacks from the enemies come from above, as well as from the side. Special weapon power-ups help players by upgrading their ships missiles from single shot to double shot, and from double shot to quadruple shot. However, be careful...shooting a power-up will cause you to loose all of your upgrades.

Only the most elite players will find themselves in possession of a Bonus medal at the end of their round. How well do you rank?

Gold Medal

Gameplay Screen -

Player Score: 0   Level 1   High Score: 150
-Status Normal-
Safe Zone
LEVEL COUNT
PLAYER SCORE
WEAPON STATUS
PLAYER FEEDBACK
HIGHEST SCORE
MISSILE UPGRADES
SIDE MISSILE UPGRADES
BONUS SHIP DROP
FORCE FIELD DROP
ENEMY SHIPS
SAFE ZONE
SAFE ZONE FEEDBACK
REMAINING SHIPS
Missed Enemies: 31
MISSED ENEMY COUNT
Safe Zone ACTIVE

Enemies and Weapons -

There are 6 different types of enemies within the game and each have a different point value assigned to them. The last two aliens are so rare only outlines of them and their point values are known. Can you last long enough to discover them for yourself?

- 25 pts.
- 50 pts.
- 100 pts.
- 150 pts.
- 250 pts.
- 500 pts.
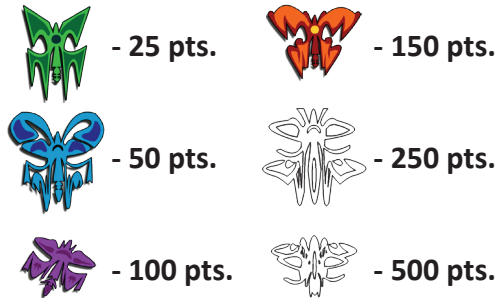
There are two missile up-grades you can attach to your space ship. Each power up has two phases of power. Switching from one type to the second type will reset you to phase one. Shooting either will remove all power ups from your ship

Phase 1   **Phase 2**
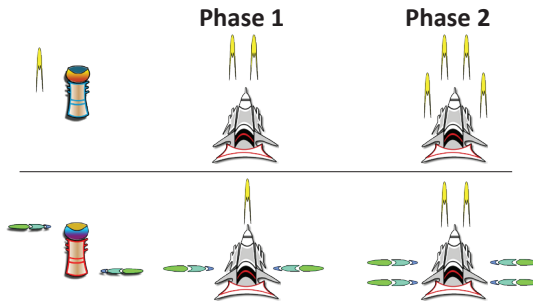
Special Bonus items -

In addition to the two types of weapon power ups that will appear on the screen there are two additional bonus items that may appear during your game.

The Shield power up icon will equip the Defend-er-K1 with a force field. This force field will allow you to collide with a single enem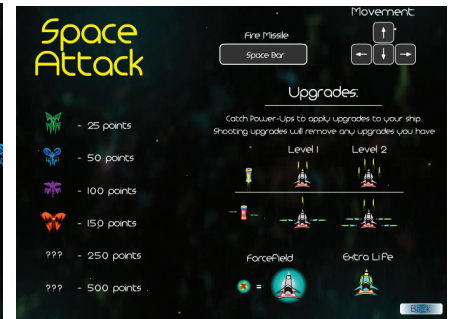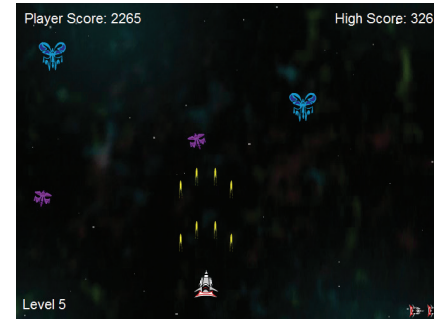y without losing a life. There is no time limit on this upgrade, but it can be destroyed if it is shot with a missile. De-stroying it will remove all of your power ups.

In addition to the Shield power up, you may en-counter a bonus ship icon. This power up is the only power up that cannot be destroyed by shooting it. If you can catch this drop you will earn a bonus life.

In addition to the power ups during the game, you may earn a special medal at the end of the game based on your performance. If your hit percentage is high enough you may see one of these medallions on the Game over Screen. Good luck and happy shooting!

Shield   **Force Field**

=

Extra Life

Gold Medal
Silver Medal
Bronze Medal
Platinum Medal

Screen shots -

Player Score: 2265   High Score: 3265
Level 5

Space Attack
Movement
Fire Missile
Space Bar
Upgrades:
Catch Power-Ups to apply upgrades to your ship. Shooting upgrades will remove any upgrades you have
- 25 points
- 50 points
- 100 points
- 150 points
??? - 250 points
??? - 500 points
Level 1   Level 2
ForceField   Extra Life

Player Score: 3325   Level 4
-Dual Sides Activated-
Force Field Enabled   High Score: 9475
Missed Enemies: 17   NO SAFE ZONE - Force Field Enabled

Game Over
Try Again?
(Press Enter to replay)
Hit / Miss Ratio:
78 Hit vs. 12 Missed
Hit Percentage: 86%
Your status: Incredible
Final Score: 2450
New High Score!
Gold Medal

Space Attack © 2013 Chad Fillion