# PixHell

## Summary

PixHell is a retro themed 1980s look and feel game where the player ship battles through space facing enemy ships along the way. It is an in-house game built using native Android SDK and Amazon Mobile App SDK and no external game engines and currently side loadable into Kindle Fire devices.

## Features

1. AI Manager  
   The AI Manager is the crux of the game engine where it generates random paths for enemy ships, in game coins, pickups.
2. Player Ship Controls

The player ship is controlled through accelerometer changes on the device. The player tilts the device in the direction he wants the ship to move. The accelerometer sensitivity is configurable through the Settings Menu which is persisted through Shared Preferences

1. Assets  
   The assets consists of custom images created through Gimp, a background music composed from scratch and SFX generated through [www.bfxr.net](http://www.bfxr.net) , which doesn’t have any licenses associated but state that “You have full rights to all sounds made with bfxr, and are free to use them for commercial purposes”.
2. Store and Inventory  
   The inventory consists of consumables which can be purchased through Amazon In-App Purchasing API or through In-Game coins which are available as pickups when some of the enemies are killed. Currently “Ship Health” is purchasable as an IAP Item for $0.99 (Tested using SDK Tester). However there is scope to expand this to “Ship Lives”, “God Mode” and similar powers. The In-Game coins could be used to buy special weapons like “Missiles”, “Bombs” and “Nukes”. For now we have only the “Missiles” implemented.
3. Cheats  
   We have cheat codes as part of our store wherein the player can submit cheat codes to get “FREE” Health packs. The idea behind this is to identify bugs within our application – any player who identifies a valid bug which has been verified by us would get a cheat code which could be used to get 2 Free Health Packs worth $1.98. However we currently do not have a server setup to maintain a cheat-store to be implemented. Hence we’ve implemented it using static strings within the application. A cheat code once used is shredded and cannot be used again. “CHEAT01” to “CHEAT05” is the cheat codes which are static within our implementation.
4. Insights SDK

## Amazon APIs Used

* Amazon In-App Purchasing API  
  This API is to be used purchase consumables like “Health” and “Lives”. In the current version of our implementation only “Health” is implemented.
* Amazon Insights SDK

## Future Features

* GameCircle API Integration

We haven’t currently worked on calculating a score for the player as he plays the game. But we plan to implement calculation of scores, and having it synced up with Game Circle API to have leaderboards and achievements.

* Whispersync Integration

Our persistence is currently done on a device basis, but giving users the option of carrying their purchases and scores across devices would be good and hence Whispersync is on the list of APIs which we would want to implement.

* Game Inventory and Consumables

Implementing more items in the inventory as mentioned earlier like having “Bombs”, “Nukes” and “Lives” as IAP Consumables would give more variety to the player to choose from.

* Amazon Coins  
  Our current In-Game coins could be extended to syncing with Amazon Coins with a factor of 100:1 (100 In-Game Coins = 1 Amazon Coin). This would expand the avenues of benefits the user has from playing the games as there are tons of things to be bought with Amazon Coins and the player retention would be good.