DADS CRC CARDS

Legend:

BLUE = Functionality Classes Green = Friendly Object Classes

MainActivity.java

X-coordinate at a constant

Keep track of it's position and

velocity

velocity

Yellow = Weapon Object Classes Red = Enemy Object Classes

Responsibilities: Collaborations: MissileCommand.java to Start the game when the app is opened start the game. Pause the game when game Hornet.java Resume paused game **Collaborations:** Responsibilities: Spawn new hornets when prompted to Hornet's by MissileCommand.java information is Store and update its coordinate accessed by the game loop to location Destroy itself once it gets to its final check for collision destination (cow location) between either a MissileCommand.java Increase its velocity when certain level cow or a missile thresholds are reached explosion. Responsibilities: Hornet calls **Explanation:** Main logic loop of the game Hornet.java is the class that tracks the hornet Cow.kill() when it Keep score of the game stingers coming down from the sky, in place collides with it. Responsible for constructing all for enemy missiles in the classic game. It flies other objects, hornets, cows, towards the cows in a straight line. It draws powerups, etc. itself, kills itself when interacts with missiles Stores hornets, cows, powerups or cows, and increases in speed each round. Collision detection (Detects when hornets touch missile explosion or cows, calls appropriate methods) Start new game Change rounds/levels Draws Cow.java - Draws the initial level screen (Foreground and Background) **Collaborations:** Responsibilities: - Also calls other draw functions Cows are spawned at the start of the If a stinger for all objects on screen, ie game or on each new level reaches a cow it hornets, cows, etc Stores its location is destroyed. Tracks FPS for animations Store its state (alive or dead) The collision Have a functionality to "moo" if user detection is done taps on the cow PowerUp.java MissileCommand. java Collaborations: Responsibilities: Keep track of what type of The game loop checks if power up it is (laser, extra user's missile explosion intersects with the power ammo, etc) Fall down from the sky along up and then the user's Building.java the Y-coordinate and constant Base.java is granted a new

Collaborations:

Same as

Cows.java

Responsibilities:

Same as Cows.java, just a different image

Missile.java

Responsibilities:

- Travel in a straight line with constant velocity from the base to user's tapped coordinate location, explode once tapped location is reached.
- Store it's location and shape
- Store it's explosion radius

Collaborations:

Game loop reads in user's OnTouchEvent and calls Base.fire() function which calls Missile.java's constructor.

It's member variables are accessed by the game loop to check for collisions

Collaborations:

- Base.java
- Cows.java
- Building.java
- Hornets.java
- Missile.java

power up, ie increase

Base.ammo.

PowerUp.java

Base.java

Responsibilities:

- Base is where the missiles are fired from
- Holds the missiles in an ArrayList
- Keeps track of what type of missiles it has and spawns the correct type
- Removes the missiles once it gets to its location

Explanation:

Base.java is the class that handles the player's DADS base. The main job of Base is to create instances of Missiles that head towards the spot where the player presses the screen, and keep track of how many Missiles they have left. It also draws itself and stores its own coordinates.

Collaborations:

Missile.java when fire() is called, passes origin and target coordinates and creates new instance of missile.