# Design Rationale – Edited for Assignment 3

Name of the project – Assignment 3.

Team name – PJs.

Team members – Joleen Tang, Pasan Jayanetty.

For assignment 3, our team had to implement two new dinosaur species to the game. Further, the team had to create new behaviours and actions as required. Due to these additions some design changes were made in order to enhance the quality of the overall design of the project. This rationale mentions the significant design features added by the team.

The following enums were added to the project:

* DinoCapabilities
  + Contained different entries that give dinosaurs additional capabilities depending on their own species type or other factors such as pregnancy.
* FoodTypes
  + An enum that differentiates Food Items into herbivores or carnivores.

Purpose of creating enums:

* Compared to constants, enums are able to group similar items in a set.
* Ability to restrict input parameters in the constructor. Which effectively reduces the information the user should know before creating a new object.
* Increases readability in code.
* Enums are iterable if needed.
* Thread safety benefits.

EatBehaviour and HuntBehaviour were completely refactored to utilize enums as opposed to constants and new objects for each iteration. This also reduced dependencies with other classes.

While implementing code, the team decided to replace public methods and attributes to private and protected methods and attributes where possible.

* Private methods and attributes ensure restricted accessibility to sensitive information within a class.
* Protected methods can only be used by classes that inherit the class where protected is situated in.
* Hence, changing public to one of private or protected significantly increases encapsulation in the assignment coding as it protects methods and attributes from being accessed and changed outside their respective classes.
* Getter and setter methods were added so that private attributes could stay private

The ‘dinosaurs’ package was created to group all the different Dinosaur species together as they were essentially just a collection of constants that were passed into the Dinosaur superclass constructor. Now the Vending Machine is the only object that directly accesses the dinosaur subclasses which increases the encapsulation.

Flying was implemented as a capability for extensibility and ease of checking; instead of typecasting, one can just check for the canFly capability.

The original main method in Application was split up into smaller functions to allow for a more complex game driver without repeated code. Initialising the game is done before each time asking for the player to choose a game mode which resets the player and map to be the same every time.

Quitting the game was implemented as an Action so that the player could access it while the game is running every turn and allows the passing of time to be recorded (through ticking every time a player turn occurs).