Written by Rob Shewaga
2018.02.26
robshewaga at gmail dot com

Pages: 7

## Highlighting Notes

Scene Game Objects are **green**.
Scripts are **blue**.
Script functions are **yellow**.
Script variables are **orange**.

## Coding Notes

Every script begins with a description of it's main usage.
Every enum or class has a comment describing it's use.
Most every variable has a comment describing what it's used for.

# Level Select

## Scripts

Attached:

- **DialogueManager**
- **LS_LevelButtonHandler**
  - **LevelStars**: This is the list of sprites to use the level star ratings. 0, 1, 2, 3 stars.
  - **LevelBoxes**: This is the list of sprites to use for the level button boxes. 0=unavailable, 1=available, 2=complete.
- **LS_LevelSelectHandler**
  - **IntroConversation**: This is the Level Select scene's intro conversation. Notice how the first line of dialogue contains the **OutFlag** LS_ANIM_HIDERIGHT. This is what makes the Assistant move to the right of the screen to grab the whiteboard.

## Levels

Attached:

- **LS_Levels**
  - **AutoPopulate**: If checked, the scene will populate the internal **Levels** list with the **Level**'s child Game Objects.
  - **Levels**: Contains all of the game's levels.

Children:

Each child represents a different level in the game. The order of children in the scene determine the ordering of the levels. If you want to add more levels, simply duplicate one of the existing ones or add a blank child and attach a **Level Input**. If more than 6 levels exist, the whiteboard will automatically show the arrows to switch between sets of 6 levels. For example:

**L3 - TestLevelBoth**

This is a level containing both a Food Drop portion and Food Quiz portion.

## PassedLevelInfo

This is the Game Object which is used to pass **Level Input** information from the Level Select scene to the Food Drop or Food Quiz scenes. When those scenes load, they try to find this object and load the level's information. If they don't find this (because you play one of those scenes in the editor instead of starting from the Level Select scene), they will use the **TestLevelInfo** in their own scenes.

# Food Drop

## Scripts

Attached:

- **DialogueManager**
- **NutrientSpawner**
- **ND_GameController**: Handles the general flow and overarching functions of the Food Drop level.
- **ND_RobotHandler**: Handles keeping track of the robot's list of wanted foods, updating the **ProgressBar** and running the robot's animations based on his movement speed.
- **ND_LevelTimer**: Handles showing/hiding/tracking the level timer, if used.

## TestLevelInfo

This is the **LevelInput** used when running the scene directly in the editor without first running the Level Select scene.

## UICanvas/ProgressBar

Attached:

- **ND_ProgressBar**
  - **WaveHeight**: The height +/- of the progress bar's subtle up/down wave animation.
  - **WaveSpeed**: How fast the animation wave happens.
  - **TransitionTime**: When the progress bar needs to be updated because a food was eaten, this is how long in seconds it takes to LERP from the old to new fill position.

## LevelInput.FoodDropLevelInput

These variables control the Food Drop game.

**Dialogues**: The list of **Dialogues** to use.

**SpawnerInfo**

- **SpawnGroupSize**: The spawner generates a randomized list of foods to spawn in groups of this size. When all these foods have been dropped, a new list spawns. Having a smaller size gives you more control over how quickly your percentages below take effect.

**LogFoodPercent**: Of the **SpawnGroupSize** of items, what percentage of those are **LogFoods**. A high value would make an easy level.

**NeededFoodChance**: When a **LogFood** spawns, the percent chance that this food is what the robot currently needs. A high value would make an easy level.

**NotFoodPercent**: Of the **SpawnGroupSize** of items, what percentage of those are **NotFoods**. A high value would make a hard level.

**NeedAllLogFoods**: Whether or not the robot needs at least one of each of the **LogFoods**. This ensures that all of the level's **LogFoods** will be added to the food log.

# Food Quiz

## Scripts

Attached:

- **DialogueManager**
- **FQ_FoodLineupHandler**: Handles the lineup of foods to drag from the bottom of the screen. The lineup can handle any number of foods.
- **FQ_GameController**: Handles the general flow and overarching functions of the Food Quiz level.

## TestLevelInfo

This is the **LevelInput** used when running the scene directly in the editor without first running the Level Select scene.

## LevelInput.FoodQuizLevelInput

These variables control the Food Quiz game.

**UseFoodDropLogFoods**: If checked, the Food Quiz's foods will be the foods listed in **FoodDropLevelInput**.**SpawnerInfo**.**LogFoods**. This is useful if the level is both a Food Drop and Food Quiz and you don't want to explicitly add the foods again to Foods.

**Foods**: The list of foods to choose from.

**FoodOrder**: This is the list of indices into **Foods** of the quiz's correct answers. If this list is 3 items long, the Food Quiz will be 3 questions. If the **FoodOrder** is 0, 1, 2, this means that **Foods**[1] is the first correct answer, **Foods**[2] is the second, and **Foods**[3] is the third.

**Dialogues**: The list of **Dialogues** to use. Each question in the quiz can be represented with a **DialogueType**.FQ_ROBOTFOODITEM to show what the robot will ask for, a **DialogueType**.FQ_FOODITEMSUCCESS to show what the Assistant will say if the question is correctly answered, and a **DialogueType**.FQ_FOODITEMFAIL to show what the Assistant will say if the question is answered incorrectly. In each of these, the **DialogueValue** is the index into **FoodOrder** of which question this dialogue refers to. The **InFlag** FQ_GLOWFOOD along with an **InFlagValue** can be used to glow one or all of the foods.

# FAQ

How do I add a level?

> Open the Level Select scene, duplicate one of the children Game Objects of **Levels** and change the **LevelInput** values. Alternatively, add an empty child Game Object of **Levels**, attach a **LevelInput**, and set the values.

How is the Food Drop scored?

> If a level is only a Food Drop portion, it is currently scored 3/3 stars and Complete regardless of whether the player's performance or win/fail overall.

How is the Food Quiz scored?

> The Food Quiz is scored based on the percentage of correct answers out of the total amount of questions. 1 correct answer out of 3 will result in 1 star.

How do we add a whiteboard flipping animation in the Level Select?

> When the sprite for the whiteboard is animating, use **CG_LevelSelect**/**Animator**/**CG**.**CanvasGroup**.alpha to show/hide the levels temporarily.

How do we add the Log Screen?

> As the Log Screen is its own scene between the Food Drop and Food Quiz levels, the main thing to make sure of is to pass on the **PassedLevelInfo** incoming from the Food Drop to the Food Quiz. This should be done automatically as the **PassedLevelInfo** will have the DontDestroyOnLoad flag set, meaning it exists between scene transitions. See **ND_GameController**.EndScene for where to insert the code to change to the Log Screen.

Why does the Level Select not remember the level completion state of more than one level at a time?

> The Level Select scene has no persistence between the different times it has been loaded. If you complete a level, it will show as complete. However, the second time you finish a level and return to the Level Select scene, the first level will not show as complete because the Level Select screen doesn't remember a player's overall progress, it only reads the previous level it just came from. Player persistence will need to be added however you like (Unity's built-in saving, or reading/saving out to a file, or creating a new Game Object for player data that doesn't get destroyed between scenes).