**About OldBirds:**

**Technical Documentation**

This document provide short description on the available classes.

The document needs to be updated after code modification.

**In Assets folder:**

**AddEventHandler.cs**:

Class AddEventHandler. It handles clicks to the week area and hourly panel.

Method OnPointerClick() gets a text from weekday.

**AnimatorHelper.cs:**

Class AnimatorHelper. It is a stub that has start() method of initialization and update() method that updates once per frame.

**BusController.cs:**

Class BusController. It has an method bLeaving(). Also it has a TimeController(). There is a method Start() that sets bLeaving() false. Update() method update the current time.

**BusDoorTrigger.cs:**

Class BusDoorTrigger. First method is Start() that is used for initialization. It starts with the game object being inside the bus. Method update() is called once per frame. OnTriggerEnter() sets bird inactive and hides it and then it adds it to onboard list.

**BusMouseOver.cs:**

Class BusMouseOver. There is a Start() method that sets busPanel false. Then there is an update() method that if door is triggered it updates it and changes method back to false. OnMouseOver() sets busPanel object true and creates items. OnMouseExit() method destroys items and sets buspanel object false. ResetOnBoardUI() method destroys items and creates new items. DestroyItems() method destroys all items. CreateItems() method creates items on the bus panel.

**ChangeEventHandler.cs:**

Class ChangeEventHandler. It has a method OnPointerClick() that gets time controller and event attributes.

**EventAttributes.cs:**

Class EventAttributes. The class only creates public bird event data.

**ShowHide.cs:**

Class ShowHide. It has a method HideOrShow() that hides or shows objects. Next method is hideOrShowSelf() that hides or shows self.

**These are all scripts in the script folder:**

**Script Files in Camera folder:**

**OrtoCamScript.cs:**

It is the main camera script of the project. It consists of public class OrtoCamScript. And has a methods start() and update() on it. Update includes keyboard scrolling, mouse scrolling, horizontal camera movement, vertical camera movement and zooming.

**SmartGlassesScript.cs**:

SmartGlassesScript has one class called SmartGlassesScript. It has a methods start() and update(). Start() sets Marjattas path and sets her moving. Marjatta will start movement routine based upon the scene name. Update() updates once per frame. There is also possibility of moving towards objects. There is also a method called RotateTowardsObject() This is used for Marjatta when she is rotating towards objects. Another method is MoveTowardsObject(). This is used for moving towards an object. GetNextWaypoint() method is meant to find the next waypoint if that is possible. OuluCityRoutine() method hides marjatta in the beginning to make it look like she comes from the house. After two seconds Marjatta is shown. MarjattaGoesPirkkoRoutine() method sets Marjattas waypoint to Pirkkos house and sets the movement speed of Marjatta. StockmannRoutine() does not do much it just let’s Marjatta wait. ContinueMoving() method sets Marjatta moving again. AnswerPhone() generates a random time for Marjatta to answer the phone. TurningAngle() method determines whether the next waypoint is on the right or the left.

**Script files in the controllers folder:**

**MainController.cs:**

Includes class MainController. It has a method start(). It start Marjatta on the path to stockmann. Marjatta is assigned to the main controller. It then starts the movement routing according to the scene name. Method Update() updates Marjattas movement path. RotateTowardsObject() method rotates Marjatta towards an object. MoveTowardsObject() method moves Marjatta towards an object. GetNextWaypoint() methods parses the current waypoints by number and adds one to it. Than it searches for a waypoint. OuluCityRutine() hides Marjatta so she can appear from her house than it waits two seconds and makes Marjatta visible. MarjattaHouseRoutine() method waits a second then starts movement towards a waypoint. MarjattaGoesPirkkoRoutine() is similar it waits a second and than starts movement towards first waypoint and sets movement speed. StockmannRoutine() starts movement towards stockmann. newSceneRoutine() does the same thing. ContinueMoving() method lets the caregiverG\_UI fall back to main menu and otherwise does the same thing as previous ones. AnswerPhone() method generates a random time to answer the phone.

**TimeController.cs:**

Its purpose is to control time in the simulation and the calendar. It has a class TimeController. It includes speed that time changes, time itself, leap years are enabled and multithreaded alarms are enabled. There is start() method to be used for initialization. Update() method is called once per frame. It needs to be updated to the current day. There is also a setSpeedForObjects() method that sets movement speed. UpdateEvents() method that updates events. UpdateEventList() method updates event lists. UpdateCalendar() method updates the calendar days when scrolling from week to week. It also initiates the day texts. CreateEventPanel() method creates an event panel. The panel contains hour list. PausetTime() method pauses time and RsumeTime() method resumes time. ForwardTime() method forwards time. ReverseTime() method reverses time to be backwards. DoubleReverseTime() method doubles the reverse speed. SetGameSpeed method sets game speed. GUINextWeek() method with GUIPrevious() week and GUIThisWeek() are GUI calendar button functionalities. ChangeEventClick() changes the event and AddNewEventClick() adds new event. AddNewEvent() adds an event and ChangeEvent() changes it. CancelButtonClick() cancels and removes deletion mark. DeleteButtonClick() deletes an event. GetCurrentDateTime() gets current time.

**Scripts in the Gizmo folder:**

**DrawArrow.cs**:

It has a class DrawArror. The class has methods ForGizmo() that draw the arrow and ForDebug() that debug the arrow.

**SimpleBox.cs:**

It has a class SimpleBox and a method OnDrawGizmos() it sets the gizmo color and a rotation matrix.

**Scripts in the GUI folder:**

**CaregiverGUIScript.cs:**

The class in this file is CaregiverGUIScript. The first method is start() which in the beginning randomizes caregiver avatar and starts a new arraylist. Update() method updates and hides left hand side UI. ShowGUI() method shows GUI if state is true and if state is false it won’t show the GUI. Method OnGUI() will give Marjatta a speech bubble when calling. It will draw the speech bubble and draw the text message received in the bubble. It will also show the caregiver GUI if it is on. If caregiver GUI is not on it will not be shown. It will also show currently received messages. After the texting has been stopped the bird will continue walking. The method will also show senior GUI if it is on. There is a way to notice text message if senior has a message. It will show currently sent and received messages.

**NavigationGUIScript.cs:**

That has a class NavigationGUIScript. It is used for user to navigate scenes manually. It has a method Update() to update whether key has been pressed or not. It also has a method OnGUI() that creates background, buttons and skip scene button for the GUI.

**Next folder is Misc:**

**AlphaFlashingScript.cs:**

That has a class AlphaFlashingScript. It is just a template to do things it has two methods Start() and Update() that do nothing.

**SelectionTool.cs:**

That has a class SelectionTool. Selection tool is for selecting objects for example for multicall purposes. This is used with the main controller gameobject. It will select gameobjects with the tag bird. It has a start() method that creates an arraylist selected birds. It also has an Update() method. Update method will accept only transforms that are tagged bird and check if the object is already found in the list. It will close info window. It will add to list if the object is not already there. It will then change shader for the silhouette. It will also remove from list and remove silhouette. The class also has a method addToSelectionList() which will add selected birds to list. It also has a method UpdateBirdList() that will update the list. And last it has a showInfoWindow() method that will show the updated list.

**The next folder is movement folder:**

**ChangeSceneTrigger.cs:**

With a class ChangeSceneTrigger. The first method OnTriggerEnter() will change the scene when there is a trigger. OnDrawGizmos() will draw a gizmo cube.

**MovementController.cs:**

It contains MovementController class. It is meant for controlling movements in the project. It has a method GetDefaultMoveSpeed() that returns the default movement speed. It also has a method GetDefaultTurningSpeed() that returns the default turning speed. It has a method Awake() that awakens the animation and sets random waypoints, visited waypoints and movement speed. Update() method update whether or not waypoint is next or last waypoint. It also makes objects move. MoveTowardsObject() moves towards objects and waypoints. If movespeed is above zero it moves towards next waypoint. If movement speed is less then zero it moves towards last waypoint. If the object is near enough of a waypoint it gets another waypoint from the list. There is also a way to randomly generate way points to keep the movement going. SetupRandomWayPoints() is a method that sets up a list of waypoints. SetNextRandomWaypoint() method sets the next random waypoint. It gets random way point from the list. RotateTowardsObject() method find a vector pointing from the position to the target. Creates rotation where the bird needs to be to look at the target. Than it rotates the bird overtime with required speed until it is in the required rotation. SetNextWaypoint() method parses the current waypoints and adds one to it. This is used for determining paths.

**InformationTextScript.cs:**

It has a class called InformationTextScript. It has a method called Start() that starts initializes information window for the old birds animations. Than it has a method show() that either enables or disables the window. It has a method setShow() that enables show. It has a method enableWindow() that enables information window and it has a method disableWindow() that disables it.

**PointOfInterest.cs:**

It has a class PointOfInterest. It has a method Awake() that awakens imagepanel, Gui, text panel. It has a method Update() that updates camera positioning. And it has a method Show() that plays audio attached to game objects and adds the name of the sound with it. It also has a method hide() that sets the instance to be false.

**WaypointScript.cs:**

It has a class called WaypointScript. It works as adding waypoints to the ground. It is dummy script that has as its main point to show Gizmo.

**Next folder in scripts is web:**

**BirdEvent.cs:**

It contains class BirdEvent. Method EventID() sets even id and returns it. StartTime() method sets start time and returns it. EndTime() method sets end time and returns it. Date() method sets date and returns it. Location() method sets location and returns it. Person() method sets person and returns it. Activity() method sets activity and returns it. Reminder() method sets reminder and returns it. BirdEvent() method doesn’t do anything.

**SQLHandler.cs:**

It has a class SQLHandler. It has a start() method that initializes events and form. It has an updateAllEvents() function that updates all events. It has AddEvent() method that adds a new event for the bird. It has an addPendingNotofication() method that adds a new pending notification. It has an changeEvent() method that allows one to change events. It has an updateEventForBird() method that allows for updating events for a single bird and then there is a deleteEvent() method for deleting events. It has an AccessDatabase() method that handles database working.

**These are scripts in the cCalendar folder:**

**CalendarDemo.cs**:

It has a class calendarDemo. Method start() intializes the caldendar. It uses the location of the current calendar display and location of time settings. Speed default is fifteen minutes. It also initializes the current camera settings and start the calendar. It also has a method called update() that updates once per frame. Another method is OnGUI() that displays the time.

**cAlarm.cs:**

It has a class cAlarm. There is a method m\_Method() that makes the alarm go off. It also has a method name() that returns the name of the alarm. It has a method cAlarm that sets name for the alarm and sets the alarm.

**cAlarmList.cs:**

It has a class cAlarmList. It has a method cAlarmList that initializes a new instance of it. AddAlarm() adds an alarm to the list. Launch() method begins execution of the thread. Abort() aborts the execution of the thread. Run() updates the list of alarms. HasAlarm() method determines whether the instance has an alarm with a name. RemoveAlarm() method removes an existing alarm.

**cCalendar.cs**:

There is a class cCalendar. This class keeps track of in-game time. It allows users to set the speed at which it changes during each update step. \_createObject() method creates object that updates each frame if it hasn’t been created already. init() method initializes the object that updates the time and loads the last saved time from the player preferences if needed. Load() method loads the saved in-game time from the player preferences and sets that as current in-game time. Method save() saves the current in-game time to player references which can then be loaded. getEndOfMonth() method gets the number of days in the current in-game month. Method getDaysInMonth() gets days in the month. getMonthName() gets the name of a custom month. getMonthabb() method gets abbreviation of the custome month. addMonth() method adds a month. ClearMonths() method removes all months. useDefaultMonths() sets default months for use. pause() method pauses the time. resume() resumes updating the month. setDay() method sets the in-game day if it is within limitations. setHour() method sets the hour if it is within limitations. setMinute() sets the minute if it is within limitation. setMonth() sets month if it is within number of months in one in-game year. setSeconds() sets the current in-game seconds. setSpeed() sets the speed that the in-game time changes. setYear() sets the current in-game year. setAlarm() sets an alarm. enableMultiThreading() enables multithreading for alarms but can cause desynchronization of time and dates. disableMultiThreading() disables multithreading. updateSeconds() updates in-game seconds. updateMinutes() updates in game mintues. updateHours() updates in-game hours. updateDays() updates in game days if there are months in the calendar. updateMonths() updates months if there is any. There is also method Awake() and start() that checks if any months are added. Update() method update the time and sets the change in time determinant to the speed of change.

**cDate.cs**:

It has a method year() that will get year object. and month() that will get a month and day() that will get day. cDate() initializes a new instance of the cDate class. setDay() sets the day. setYear() sets a year. setMonth() sets a month and it is thrown if a month does not exist.

**cDateAlarm.cs:**

It has a class cDateAlarm. It has a method cDateAlarm() that initializes a new instance of it. it also has a method IsAlive()that determines whether this instance is alive or not. Run() method will set of alarm if it is not already run.

**cDateTimeAlarm.cs**:

It has a class cDateTimeAlarm that sets alarm of at specific date and time. cDateTimeAlarm() method is used for initialization. IsAlive() method determines whether alarm has already gone off. Run() method sets of alarm if it hasn’t already gone off.

**cMonth.cs:**

It has a class cMonth. name() method gets the name. abb() method gets abbreviation. days() method gets the days. leapyearDays() gets days in the leap year. normalDays() gets days in a non leap year. cMonth() method initializes a new instance of it.

**cTime.cs**:

It has a class cTime. Method hour() gets hours. minute() gets minutes. seconds() gets seconds. cTime() method initializes a new instance of it. setHour() method setst he hour. setMinutes() method sets the minutes and setSeconds() method sets seconds.

**cTimeAlarm.cs**:

It has a class cTimeAlarm. Method cTimeAlarm() initializes a new instance of it. IsAlive() determines whether this alarm has already gone off. Run() sets off the alarm if it hasn’t gone off already.