**Автономное общеобразовательное учреждение**

**«Средняя общеобразовательная Школа №85**

**Индивидуальный проект по информатике**

**«Проблема ориентации по школе»**

Работу подготовил Ученик 9В класса Сокотухин Иван Проверил работу учитель информатики и ИКТ Романова Галина Владимировна

**Кемерово, 2022**

**Оглавление**

[**Введение** 3](#_Toc101372892)

[**Выбор Метода решения проблемы** 4](#_Toc101372893)

[**Разработка** 5](#_Toc101372894)

[**Немного о движке Unity** 6](#_Toc101372895)

[**Разработка основных механик** 7](#_Toc101372896)

[**Дизайн** 8](#_Toc101372897)

[**Итог** 9](#_Toc101372898)

[**Список литературы** 10](#_Toc101372899)

[**Приложение** 11](#_Toc101372900)

# **Введение**

Наша школа – самая крупная школа в Кузбассе, и поэтому у большинства родителей учеников возникает проблема с нахождением того или иного кабинета, когда они идут на родительские собрания и т.п. Мои родители также сталкиваются с этой проблемой и всегда просят меня объяснить им, где находится нужный кабинет. Такая же проблема возникает и у «новеньких» в школе, которые еще только учатся ориентироваться по школе.

Передо мной встала проблема – проблема ориентации в нашей «огромной» школе, и в моих силах её решить.

**Гипотеза исследования:** Сложность в ориентации по школе

**Объект исследования:** Помещение школы №85

**Предмет исследования:** Ориентация по школе №85

**Цель исследования:** облегчить ориентацию по школе

**Задачи исследования:**

* Найти рабочий метод решения поставленной проблемы
* Разработать основные механики
* Разработать дизайн
* Воплотить проект в «жизнь»
* Сбор информации о кабинетах

**ВОЗМОЖНЫЕ МЕТОДЫ РЕШЕНИЯ ПРОБЛЕМЫ:**

* Создание буклета с подробной картой школы
* Создание приложения с картой школы и информацией о кабинетах

# **Выбор метода решения проблемы**

**Буклет:**

**Плюсы:**

* Не требует гаджетов
* Для некоторых людей этот метод удобнее

**Минусы:**

* Большой объём самого буклета при небольшой информации, которую он несёт
* Бумага может порваться или испортиться, или потеряться

**Приложение:**

**Плюсы:**

* Всегда под рукой
* Не потеряется, в отличие от буклета
* Несет в себе большой объём информации

**Минусы:**

* Приложение нужно скачивать
* Исходя из всех вышеперечисленных плюсов и минусов, я выбрал создание приложения.

# **Разработка**

Для разработки я решил выбрать движок Unity, т. к. уже имею 3-х летний опыт работы с ним, также Unity является одним из популярнейших движков для разработки мобильных приложений

# **Немного о движке Unity**

**Unity** — межплатформенная среда разработки компьютерных и мобильных игр и приложений, разработанная американской компанией Unity Technologies. Unity позволяет создавать приложения, работающие на более чем 25 различных платформах, включающих персональные компьютеры, игровые консоли, мобильные устройства, интернет-приложения и другие. Выпуск Unity состоялся в 2005 году и с того времени идёт постоянное развитие. Основными преимуществами Unity являются наличие визуальной среды разработки, межплатформенной поддержки и модульной системы компонентов. К недостаткам относят появление сложностей при работе с многокомпонентными схемами и затруднения при подключении внешних библиотек. На Unity написаны тысячи игр, приложений, визуализации математических моделей, которые охватывают множество платформ и жанров.

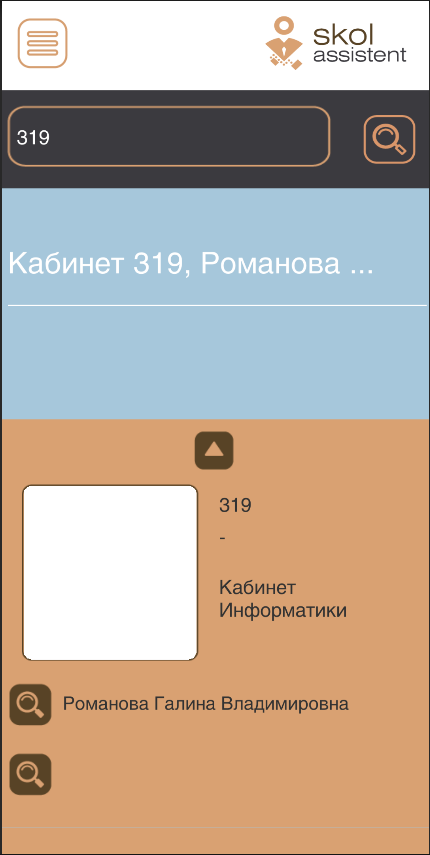
# **Разработка основных механик**

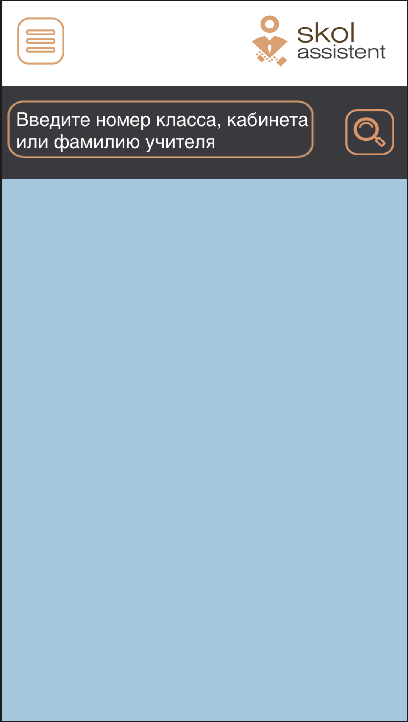
Для начала необходимо понять, что должно быть в моём приложении. В нём должна быть система построения маршрута от одного кабинета до другого, возможность поиска информации о кабинете по его номеру или другой информации. В качестве основных механик приложения мною будут использоваться следующие механики: встроенный UI (пользовательский интерфейс) для меню и окон приложения, а также для поиска кабинета и NavMesh (навигационная сетка) для построения маршрута. Также необходимо разработать механику поиска кабинета по различным параметрам

**Мною было разработано несколько скриптов, отвечающих за основные механики:**

* Код, осуществляющий поиск информации о кабинете (см. приложение )
* Код, осуществляющий построение маршрута до кабинета (см. приложение)
* Код, осуществляющий появление маркеров на карте во время построения маршрута (см. приложение)
* Код, осуществляющий всё взаимодействие с UI-элементами (см. приложение)
* Код, благодаря которому возможно перемещение камеры по карте (см. приложение)
* Код, благодаря которому можно узнать информацию о кабинете (см. приложение)

# **Дизайн**

Начальное меню Меню справочника Экран загрузки

Меню поиска кабинета Иконка приложения для тел

# **Итог**

Разработав все механики приложения, которые предусматривались мною изначально, а также дизайн и логику в Unity, я объединил это всё в программе, получив на выходе готовое приложение в формате APK для операционной системы Android. *Цели моего проекта можно считать выполненными:*

* Был создан дизайн приложения
* Были разработаны основные механики
* Была собрана информация о кабинетах
* Было создано приложение для Android

# **Список литературы**

<https://docs.unity3d.com/ru/530/Manual/>

# **Приложение**

|  |
| --- |
| public class ПОИСК : MonoBehaviour  {  public GameObject[] blocks;  public GameObject Поисковая\_строка;  public string text;  public GameObject[] включить;  public int сколько\_заполнено = 0;  public GameObject[] pivot;  int i;  int a;  int b = 0;  public void Search()  {  b = 0;  сколько\_заполнено = 0;  if (включить[0] != null)  {  for (a = 0; a < включить.Length; a++)  {  if (включить[a] != null)  {  включить[a].SetActive(false);  }  }  включить = new GameObject[5];  }  text = Поисковая\_строка.GetComponent<Text>().text.ToString();  for (i = 0; i < blocks.Length; i++)  {  if (blocks[i].GetComponent<Информацияэ>().Номер\_класса == text | blocks[i].GetComponent<Информацияэ>().Номер\_кабинета == text | blocks[i].GetComponent<Информацияэ>().Фамилия\_учителя == text)  {  blocks[i].SetActive(true);  blocks[i].transform.position = pivot[сколько\_заполнено ].transform.position;  включить[b] = blocks[i];  b++;  сколько\_заполнено++;  }  }  }  } |
| Код, осуществляющий поиск информации о кабинете |

|  |
| --- |
| public class selectdestanation : MonoBehaviour  {  public GameObject myDestanation;  bool timer\_start;  int count;  float time = 0.7f;  void OnMouseDown()  {  GameObject logic = GameObject.Find("logic");  if(count != 2)  {  timer\_start = true;  if (time > 0)  {  count = count + 1;  if (count == 2)  {  count = 0;  if (logic.GetComponent<uimanager>().istransform == false)  {  logic.GetComponent<uimanager>().dest = myDestanation;  logic.GetComponent<uimanager>().name\_of\_object = gameObject.name;  logic.GetComponent<uimanager>().sss();  if (logic.GetComponent<uimanager>().number\_of\_btn == 2)  {  logic.GetComponent<uimanager>().btn2.GetComponentInChildren<Text>().text = gameObject.transform.name;  }  if (logic.GetComponent<uimanager>().number\_of\_btn == 1)  {  logic.GetComponent<uimanager>().btn1.GetComponentInChildren<Text>().text = gameObject.transform.name;  }  }  }  }  }  }  void Update()  {  if(timer\_start == true)  {  time -= Time.deltaTime;  if(time <= 0)  {  count = 0;  timer\_start = false;  time = 0.7f;  }  }  }  } |
| Код, осуществляющий построение маршрута до кабинета |

|  |
| --- |
| public class strelochka\_spawn : MonoBehaviour  {  float time;  public GameObject Strelka;  public bool start\_spawn;  public float dist;  public float timertry;  int len;  public GameObject[] strelki;  public GameObject transform\_pos;  void Start()  {  time = timertry;  }  void Update()  {  if(start\_spawn == true)  {  time -= Time.deltaTime;  if(time <= 0)  {  time = timertry;  GameObject strelka = Instantiate(Strelka, transform\_pos.transform.position, transform\_pos.transform.rotation);  strelki[len] = strelka;  len += 1;  }  dist = Vector3.Distance(gameObject.GetComponent<NavMeshAgent>().destination, gameObject.transform.position);  if(dist <= 0.9f)  {  GameObject logic = GameObject.Find("logic");  logic.GetComponent<uimanager>().postroenie.SetActive(false);  start\_spawn = false;  logic.GetComponent<uimanager>().istransform = false;  }  }  }  public void clear()  {  for (int i = 0; i < len; i++)  {  Destroy(strelki[i]);  }  }  } |
| Код, осуществляющий появление маркеров на карте во время построения маршрута |

|  |
| --- |
| public class uimanager : MonoBehaviour  {  public int number\_of\_btn = 0;  public string name\_of\_object;  public GameObject dest;  public GameObject btn1;  public GameObject btn2;  public GameObject finish\_dest;  public GameObject start\_dest;  GameObject agent;  public bool isUI;  public GameObject postroenie;  public bool istransform;  public string bool\_name;  public string trigger\_name;  int i;  bool CAN;  void Start()  {  Application.targetFrameRate = 120;  agent = GameObject.FindWithTag("agent");  }  public void start\_pos()  {  number\_of\_btn = 1;  }  public void last\_pos()  {  number\_of\_btn = 2;  }  public void sss()  {  if (istransform == false)  {  if(number\_of\_btn == 1)  {  start\_dest = dest;  }  if (number\_of\_btn == 2)  {  if(isUI == true)  {  isUI = false;  }  else  {  finish\_dest = dest;  }  }  }  }  public void start\_transform()  {  agent.GetComponent<strelochka\_spawn>().clear();  agent.GetComponent<strelochka\_spawn>().start\_spawn = true;  agent.GetComponent<NavMeshAgent>().Warp(start\_dest.transform.position);  agent.GetComponent<NavMeshAgent>().destination = finish\_dest.transform.position;  postroenie.SetActive(true);  istransform = true;  CAN = true;  }  public void reload\_scene(string sceneNane)  {  SceneManager.LoadScene(sceneNane);  }  public void destanation\_from\_search(GameObject obj)  {  isUI = true;    if (number\_of\_btn == 1)  {  dest = obj;  btn1.GetComponentInChildren<Text>().text = obj.transform.parent.name;  start\_dest = dest;  }  if (number\_of\_btn == 2)  {  btn2.GetComponentInChildren<Text>().text = obj.transform.parent.name;  finish\_dest = obj;  }  }  public void setSibilling( RectTransform transform )  {  transform.SetAsLastSibling();  }  public void BoolSet(Animator animator)  {  if(i == 0)  {  animator.SetBool(bool\_name, true);  i = i + 1;  }  else  {  animator.SetTrigger(trigger\_name);  }  }  private void Update()  {  if(CAN == true)  {  float dist = Vector3.Distance(agent.transform.position, start\_dest.transform.position);  print(dist);  }  }  } |
| Код, осуществляющий всё взаимодействие с UI-элементами |

|  |
| --- |
| public class Camera\_Controller : MonoBehaviour  {  private Vector3 startPos;  private Camera cam;  public float speed;  float targetposX;  float targetposY;  void Start()  {  cam = gameObject.GetComponent<Camera>();  targetposX = transform.position.x;  targetposY = transform.position.z;  }  void Update()  {  if (Input.GetMouseButtonDown(0)) startPos = cam.ScreenToWorldPoint(Input.mousePosition);  else if (Input.GetMouseButton(0))  {  float posX = cam.ScreenToWorldPoint(Input.mousePosition).x - startPos.x;  float posY = cam.ScreenToWorldPoint(Input.mousePosition).z - startPos.z;  targetposX = Mathf.Clamp(transform.position.x - posX, -20f, 65f);  targetposY = Mathf.Clamp(transform.position.z - posY, -65f, 15f);  }  transform.position = new Vector3(Mathf.Lerp(transform.position.x, targetposX, speed \* Time.deltaTime),transform.position.y, Mathf.Lerp(transform.position.z, targetposY, speed \* Time.deltaTime));  }  public void Transform\_to\_cabinet(GameObject cabinet)  {  gameObject.transform.position = new Vector3(cabinet.transform.position.x, transform.position.y, cabinet.transform.position.z);  }  } |
| Код, благодаря которому возможно перемещение камеры по карте |

|  |
| --- |
| public class info : MonoBehaviour  {  [Header("Переменные")]  [Multiline]  public string FIO;  private string Klass;  [Multiline]  public string predmet;    public Texture photo;  [Header("Объекты")]  public Text FIO\_TEXT;  public Text Klass\_nomer;  public Text klass\_imya;  public Text PREDMET\_NAZVANIE;  public Image image;  private void Start()  {  Klass = GetComponent<Информацияэ>().Номер\_кабинета;  }  public void smena\_infy()  {  FIO\_TEXT.text = FIO;  Klass\_nomer.text = "Кабинет № " + Klass;  PREDMET\_NAZVANIE.text = predmet;  Klass\_nomer.text = Klass;  klass\_imya.text = GetComponent<Информацияэ>().Номер\_класса;  image.canvasRenderer.SetTexture(photo);  }  } |
| Код, благодаря которому можно узнать информацию о кабинете |