

Федеральное государственное бюджетное образовательное учреждение
высшего образования

«ОРЛОВСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ
ИМЕНИ И.С. ТУРГЕНЕВА»

Институт приборостроения, автоматизации и информационных
технологий

Кафедра программной инженерии

ОТЧЕТ
по производственной практике

на материалах АНО «Центр Интернет-образования»

Студент

Шорин В.Д.

Группа 71ПГ

Институт приборостроения, автоматизации
и информационных технологий

Направление 09.03.04 «Программная инженерия»

Руководитель практики
от университета

Чижев А.В.

Руководитель практики
от профильной организации

Забелин С.А.
М.П.

Оценка защиты

12.9.19

Орел, 2019

**Федеральное государственное бюджетное образовательное учреждение
высшего образования
«ОРЛОВСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ
ИМЕНИ И.С. ТУРГЕНЕВА»**

**Институт приборостроения, автоматизации и информационных
технологий
Кафедра программной инженерии**

Направление подготовки 09.03.04 «Программная инженерия»

**ИНДИВИДУАЛЬНОЕ ЗАДАНИЕ
на производственную практику**

для обучающегося Шорина Владислава Дмитриевича

2 курса очной формы обучения, группы 71ПГ

Место прохождения практики: АНО «Центр Интернет-образования»

Срок прохождения практики с «29» июня 2019 г. по «12» июля 2019 г.

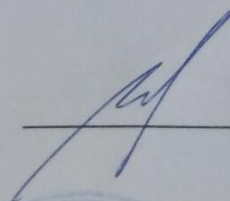
Содержание практики (вопросы, подлежащие изучению):

- описание общих сведений о имитаторе закабинного пространства;
- определение задач модернизации имитатора закабинного пространства;
- описание изменений, создание диаграмм и схем.

Планируемые результаты практики:

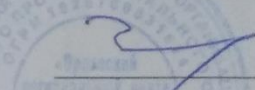
- модернизированное ПО имитатора закабинного пространства.

Руководитель практики
от университета

 Чижов А.В.

Согласовано:

Руководитель практики
от профильной организации

 Забелин С.А.
М.П.

СОВМЕСТНЫЙ РАБОЧИЙ ГРАФИК (ПЛАН) проведения производственной практики

Обучающегося 2 курса очной формы обучения, группы 71ПГ

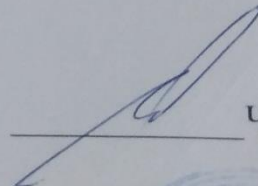
Направление подготовки 09.03.04 «Программная инженерия»

Место прохождения практики АНО «Центр Интернет-образования»

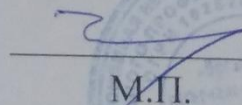
Срок прохождения практики с «29» июня 2019 г. по «12» июля 2019 г.

№ п/п	Наименование этапа проведения практики*	Вид работ	Срок прохождения этапа практики	Форма отчетности	Отметка о выполнении
1	Подготовительный (организационный) этап	1. Организационное собрание для разъяснения целей, задач, содержания и порядка прохождения практики. 2. Инструктаж по технике безопасности. 3. Выдача индивидуального задания.	29.06.2019	Индивидуальное задание	
2	Основной этап	1. Ознакомление с конкретными видами деятельности в соответствии с положениями структурных подразделений и должностными инструкциями. 2. Сбор информации и материалов практики. 3. Выполнение программы практики, индивидуального задания на практику. 4. Обработка, систематизация и анализ фактического и теоретического материала.	30.06.2019 - 11.07.2019	Фактический и теоретический материал, результаты анализа	
3	Заключительный этап	Составление отчета по практике. Защита отчета по практике с представлением материалов конкретной профильной организации.	12.07.2019	Отчет по практике	

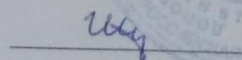
Руководитель практики
от университета

 Чижов А.В.

Руководитель практики
от профильной организации

 Забелин С.А.
М.П.

С рабочим графиком (планом) ознакомлен:
Обучающийся

 Шорин В.Д.

Содержание

Введение	7
1 Общие сведения об имитаторе закабинного пространства	8
2 Задачи модернизации ПО ИЗП	12
3 Модернизация блока редактора погодных условий	14
Заключение	17
Список литературы	18
Приложение А (обязательное) Исходный текст программы.....	19

Введение

В ходе прохождения производственной практики мною было получено задание модернизации блока редактора погодных условий имитатора местности.

Суть задания заключается в следующем: имеется программная система имитатора местности, включающая в себя блок редактора погодных условий, в котором пользователь может настраивать погодные условия и освещенность на сцене по своему усмотрению из предложенных вариантов настройки, в которой необходимо произвести модернизацию отдельных компонентов.

Целью практики является разработка и внедрение соответствующих новых функций (или изменение процесса работы старых) по модернизации блока редактора погодных условий имитатора местности.

Задачами, которые необходимо выполнить для достижения поставленной цели, являются:

- 1) выявление частей ИЗП, требующих улучшения функциональности;
- 2) проведение модернизации выявленных частей;
- 3) проведение тестирования и отладки модернизированных частей;
- 4) разработка необходимых диаграмм и схем.

1 Общие сведения об имитаторе закабинного пространства

В рамках научно-исследовательской, опытно-конструкторской и технологической работ ведётся разработка и создание всепогодного и всесезонного комплекса для обеспечения поисково-спасательных операций (КОПСО), проводимых с помощью летательных аппаратов в условиях Арктики.

Для обеспечения проведения комплексной отладки, предварительных и приемочных испытаний опытных образцов КОПСО, а также для проведения приемосдаточных испытаний серийных образцов используется технологический стенд комплексной настройки и проверки (ТСКН) [2].

Процессы комплексной отладки, предварительных и приемочных испытаний опытных образцов КОПСО, приемосдаточных испытаний серийных образцов с использованием ТСКН, сопряжены с необходимостью проведения большого количества тестовых запусков в части отладки и проверки корректности функционирования программного обеспечения аппаратуры управления и комплексной обработки информации. В условиях отсутствия (до проведения большого количества летных испытаний) достаточного количества исходных данных, получаемых с измерительных блоков КОПСО, необходимо обеспечить генерацию подобных изображений в режиме имитации пролета в определенной местности с заданными параметрами [2].

С целью обеспечения такой возможности в состав ТСКН входит ИЗП (имитатор закабинного пространства или имитатор местности), предназначенный для моделирования измерительной информации (генерации файлов различного формата) от разноспектральных датчиков, входящих в состав аппаратуры КОПСО, при пролете летательного аппарата (ЛА) над участком местности с заданными характеристиками при заданных условиях.

На основе анализа требований к назначению и области применения разработана схема программной системы имитатора закабинного пространства (ИЗП), приведённая на рисунке 1.

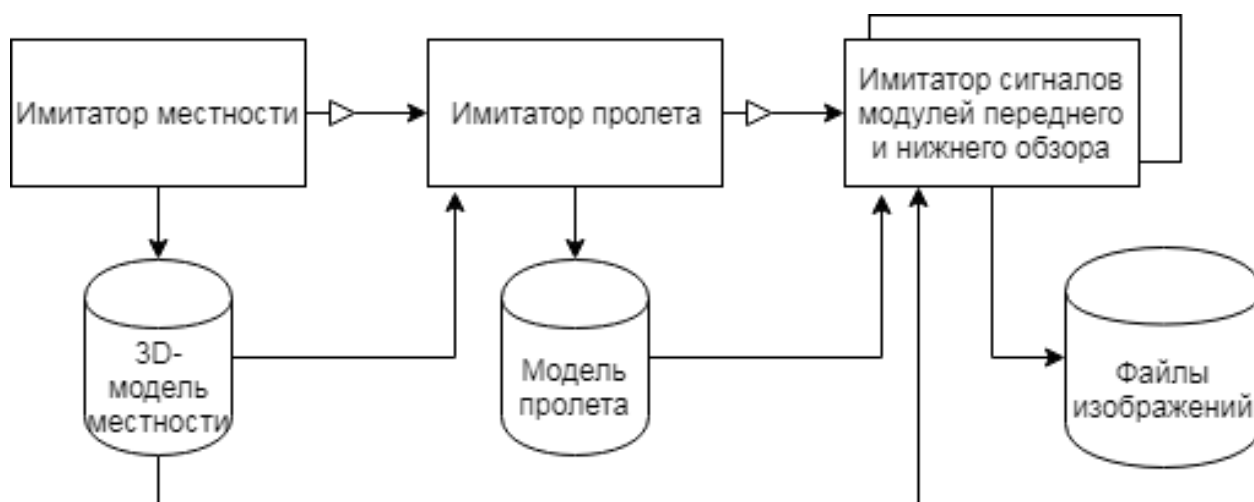


Рисунок 1 – Схема взаимодействия программ программной системы
ИЗП

Имитатор местности с объектами, расположенными не ней, реализующий 3D-модель окружающей обстановки в области наблюдения, должен обеспечить:

- 1) выбор в интерактивном режиме участка местности из набора готовых ландшафтов;
- 2) расположение на подстилающей поверхности и задание параметров различных объектов;
- 3) определение параметров окружающей среды (освещенности и погодных условий, включая температуру, туман, осадки);
- 4) генерацию трехмерной модели окружающей местности.

Исходя из этого, имитатор должен включать в себя редактор окружающей местности.

Редактор представляет собой компьютерную программу, в которой пользователю предоставляется возможность выбирать в интерактивном режиме участок местности из готовых ландшафтов и объекты с определёнными характеристиками [1].

Основными составляющими компонентами имитатора местности являются:

- 1) редактор погодных условий;
- 2) редактор местности;

3) интерфейс пользователя.

Блок редактор местности позволяет пользователю выбрать требуемую для редактирования местность, произвести настройку расположенных на ней подстилающих поверхностей, выбрать объекты, наличие которых необходимо на моделируемой местности, и разместить их согласно особенностям ландшафта, а также при необходимости сохранить созданный прототип местности или загрузить для использования или дальнейшего редактирования уже имеющийся.

Блок редактора местности можно разделить на следующие модули согласно их функциям и назначению:

- 1) модуль загрузки ландшафта;
- 2) модуль настройки подстилающих поверхностей;
- 3) модуль настройки объектов;
- 4) модуль сохранения и загрузки сцены.

Блок редактора погодных условий предоставляет пользователю возможности по настройке необходимых для имитации условий видимости и погодных условий и внесению изменений в моделируемую местность путем установки параметров для водных поверхностей и определения участков ландшафта, на которых должна располагаться снежная поверхность.

Блок редактора погодных условий включает в себя следующие составляющие модули:

- 1) модуль настройки погодных условий;
- 2) модуль настройки водной поверхности;
- 3) модуль настройки снежной поверхности.

С помощью блока интерфейса пользователь получает доступ к возможностям редактора местности и погодных условий и осуществляет управление процессом редактирования. Блок интерфейса также обеспечивает взаимодействие программы с пользователем в форме диалога.

Основные составляющие блоки и модули программной системы имитатора местности представлены в виде диаграммы компонентов на рисунке 2.

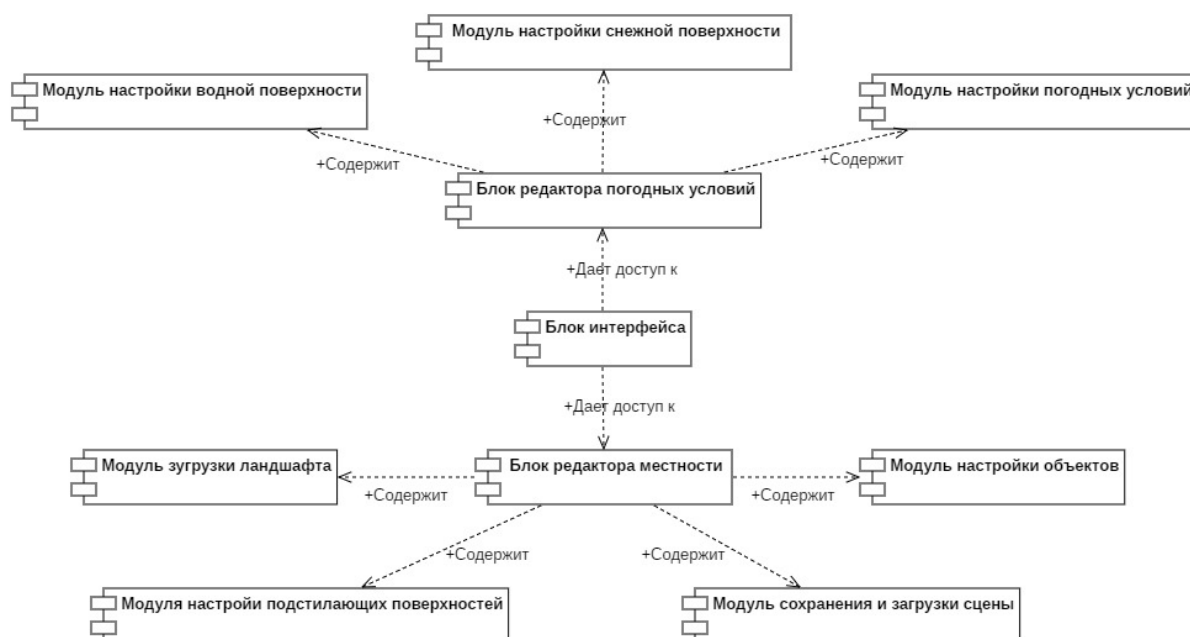


Рисунок 2 – Диаграмма компонентов имитатора местности

2 Задачи модернизации ПО ИЗП

В ходе этапа тестирования в соответствии с итерационной моделью жизненного цикла программного обеспечения был проведен анализ работы основных частей имитатора закабинного пространства, в ходе проведения которого были выделены следующие задачи модернизации программного обеспечения имитатора закабинного пространства:

В блоке редактора местности:

— для улучшения возможности перемещения по редактору сцен необходимо добавить возможность перемещения по вертикали (например, используя колесо мыши);

— для улучшения возможности перемещения по редактору сцен необходимо добавить возможность быстрого приближения к точке (т.е. масштабирования);

— необходимо добавить видимые границы зоны моделирования по длине, ширине и высоте;

— для улучшения возможности работы с объектами необходимо добавить инспектор объектов, т.е. в окне выбора объектов добавить кнопку «Инспектор объектов», при нажатии на которую в верхнем углу экрана появляется окно со списком добавленных на сцену объектов. У окна есть только кнопка закрытия. Соответственно, при добавлении пользователем нового объекта на сцену, он также появляется в списке инспектора объектов. Клик по элементу списка в инспекторе объектов перемещает камеру к соответствующему объекту.

В блоке редактора погоды:

— при включении Луны происходит только смена «скайбокса» (т.е. окружающего ландшафт видимого пространства), но не происходит автоматического понижения интенсивности освещения;

— при включении погодного условия «Туман» замечено несовпадение значение введенной интенсивности и реальной действительности, а также туман привязан к точке обзора, хотя должен быть привязан к поверхности;

— при использовании режима нанесения снега происходит нанесение снега под водой;

— при использовании режима нанесения снега при переходе кисти со льда на сушу и обратно исчезает нанесенный участок снега на суше или льду соответственно.

В остальной части программы:

— в окне настройки датчиков: значения параметров вибрации датчиков одинаковы в пределах модуля. Нужно удалить параметры «Амплитуда вибраций» и «Частота вибраций» из датчиков и вынести их на уровень модулей обзора;

— в окне настройки датчиков: «Период обзора» для каждого датчика заменить на «Период обновления информации» и «Диапазон изменения периода обзора» на «Диапазон изменения периода обновления информации»;

— в окне настроек датчиков: изменить «Конфигурационные параметры обзора датчиков» на «Параметры опроса датчиков»;

— в окне общих настроек: добавить возможность установки введенных пользователем значений параметров в качестве «значений по умолчанию»;

— добавить настройку разрешения выходного изображения для каждого датчика, соответственно, удалив блок «Конфигурационные параметры выходных изображений». По умолчанию должны быть следующие значения: изображения видимого спектр, датчиков коротковолнового и длинноволнового ИК-излучения переднего обзора – 896×672, изображения датчиков лазерного локатора и радиолокатора переднего обзора – 578×578, изображения датчиков нижнего обзора – 672×672.

3 Модернизация блока редактора погодных условий

Рассмотрим процесс решения поставленных задач более подробно. Начнем с первой обозначенной задачи и далее пойдем по порядку.

Для решения первой задачи необходимо внести некоторые изменения в скрипт «LightIntensity», а именно в методы `SetLightSource(int type)` и `SetIntensity(float intense)`. В первом методе мы добавим вызов функции `SetIntensity(float intense)` в первом блоке проверки условия (т.е. если выбран тип 1, соответствующий Луне) и передадим ей значение «50». Таким образом, если будет выбран тип источника освещения Луна, то значение интенсивности автоматически изменится на новое, равное 50. Тоже самое сделаем в следующем блоке проверки условия (т.е. если выбран тип 2, соответствующий Солнцу), передав значение «100». Таким образом, если будет выбран тип источника освещения Солнце, то значение интенсивности автоматически изменится на новое, равное 100. Изменения метода `SetIntensity(float intense)` состоят в том, что была добавлена строка

```
intensitySlider.value = intense;
```

которая присваивает полученное значение `intense` ползунку, регулирующему интенсивность освещенности. Скрипт `LightIntensity` отвечает за настройку интенсивности источника освещения и не влияет на работу других компонентов, что соответствует требованиям надежности и безопасности.

Туман представляет собой систему частиц, предоставленной средой `Unity[5]`, использующую изображение дыма, и создается путем многократного циклического отображения данного изображения на некоторой площади с определенным размером. Поэтому, при включении погодного условия «Туман» замеченное несовпадение введенной интенсивности и реальной действительности вызвано некорректными настройками основных пунктов системы частиц. Для решения данной проблемы были проведены изменения в основных настройках системы частиц (рисунок 3), а именно изменено время жизни и количество частиц.

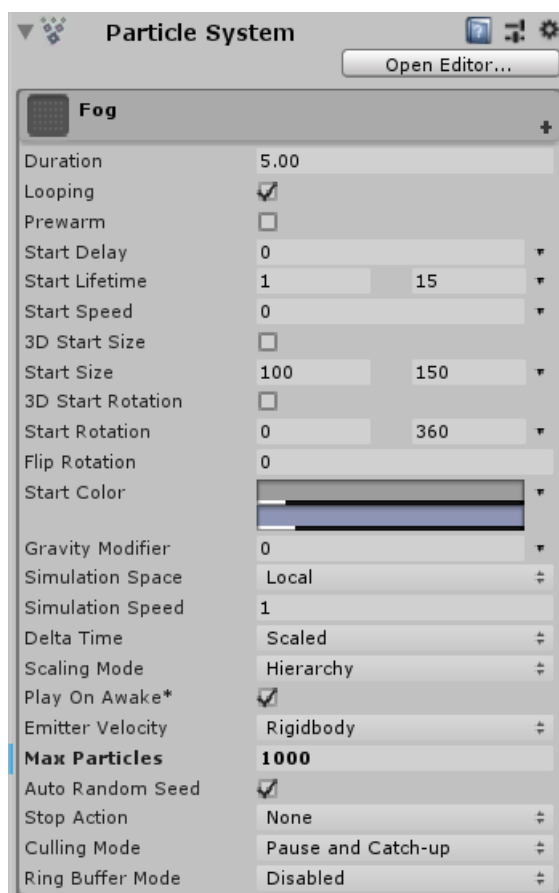


Рисунок 3 — Основные настройки системы частиц «Туман»

Для решения второй части поставленной задачи, изменим параметры привязки системы частиц к объекту: теперь сделаем его независимым от точки обзора и установим в центр ландшафта, сделав соответствующие размеры. Также, увеличим количество частиц и размер каждой частицы в целях сохранения производительности.

Последние две задачи используют одно решение, а именно проверку местности, на которую происходит нанесение снега. Если область кисти для нанесения снега попадает в область водного объекта, то там не наносится снег и сопрограмма, реализующая нанесение снега, вызывается только для области, соответствующей ландшафту.

Таким же образом решается и вторая задача. Мы делаем проверку на область, в которой происходит нанесение снега и для каждого объекта вызываем сопрогammu нанесения снега, передавая соответствующие координаты[4].

Ниже представлен фрагмент кода, реализующий данные проверки и вызов сопрограмм.

```

if (Input.GetMouseButton(0) && snowSetting && mode
==EditorMode.snow_setting){
    if (EventSystem.current.IsPointerOverGameObject()) { return; }

    Transform objectHit = rayHit.transform;

    RaycastHit downRayHit;
    Physics.Raycast(rayHit.point - new Vector3(0, 1f, 0), Vector3.up, out
downRayHit);

    if (objectHit.GetComponent<Terrain>()) {
StartCoroutine(terrainManager.GetComponent<SnowTerrainGenerator>().SetSno
w(rayHit.point, false));
    }
    else
        if (objectHit.GetComponent<IceScript>()) {

StartCoroutine(terrainManager.GetComponent<SnowTerrainGenerator>().SetSno
w(rayHit.point, true));
        } else
            if (downRayHit.transform.GetComponent<IceScript>()) {

StartCoroutine(terrainManager.GetComponent<SnowTerrainGenerator>().SetSno
w(downRayHit.point, true));
            }
        }
    }
}

```

Заключение

Поскольку части имитатора закабинного пространства, требующие модернизации, выявлены, проведено их улучшение, а также осуществлено тестирование и отладка, построены необходимые диаграммы и схемы, а также соответствуют поставленной цели, поставленную цель практики можно считать достигнутой.

В ходе прохождения практики были закреплены знания по языку программирования C# и межплатформенной среде разработки компьютерных игр Unity.

Список литературы

1. Техническое задание на «Программный имитатор закабинного пространства» от 30.08.2018
2. Программный имитатор закабинного пространства для применения в составе технологического стенда комплексной настройки и проверки комплекса для обеспечения поисково-спасательных операций, проводимых с помощью летательных аппаратов в условиях Арктики: Пояснительная записка УРКТ. 04.08.01-01 81 01 ЛУ
3. Программный имитатор закабинного пространства для применения в составе технологического стенда комплексной настройки и проверки комплекса для обеспечения поисково-спасательных операций, проводимых с помощью летательных аппаратов в условиях Арктики: Описание программы УРКТ. 04.08.01-01 13 01 ЛУ
4. Шилдт, Г. С# 4.0: полное руководство [Текст] / Г. Шилдт. — ООО «И.Д. Вильямс», 2011. — 297 с.
5. Дикинсон, К. Оптимизация игр в Unity 5. Советы и методы оптимизации приложений [Текст] / К. Дикинсон. — ДМК-Пресс, 2017. — 306 с.

Приложение А

(обязательное)

Исходный текст программы

«WeatherEditor.cs»

```
using System.Collections;using System.Collections.Generic;using UnityEngine;using UnityEngine.UI;
public class WeatherEditor : MonoBehaviour {
    public GameObject snow;
    public GameObject rain;
    public GameObject fog;
    public Slider intensitySlider;
    public ParticleSystem currentPS;
    public float userIntensity;
    public float psIntensity;
    GameObject[] weatherObjects;
    void Start () {    weatherObjects = new GameObject[3] { snow, rain, fog }; }
    public void SetActive(int weatherId) {
        for (int i = 0; i < weatherObjects.Length; i++) {
            if (weatherId == i) {
                weatherObjects[i].SetActive(true);
                currentPS = weatherObjects[i].GetComponent<ParticleSystem>();
            } else {
                weatherObjects[i].SetActive(false);
            }
        }
    }
    public void SetIntensity() {
        if (currentPS == null) return;
        var emission = currentPS.GetComponent<ParticleSystem>().emission;
        emission.rateOverTime = Mathf.RoundToInt(intensitySlider.value * 10);
    }
    public void SetIntensity(float intense) {
        if (currentPS == null) return;
        var emission = currentPS.GetComponent<ParticleSystem>().emission;
        emission.rateOverTime = Mathf.RoundToInt(intense * 10);
    }
}
```

«LightIntensity.cs»

```
using System.Collections;using System.Collections.Generic;using UnityEngine;using UnityEngine.UI;
public class LightIntensity : MonoBehaviour {
    public GameObject editLight;
    public Slider intensitySlider;
    public InputField posXField;
    public InputField posZField;
    public Material skyMaterial;
    private Color skyColor;
    public GameObject moon;
    public Material materialMoon;
    public Material materialSun;
    public float posX, posY, posZ, radius;
    public void SetIntensity() {
        float currentIntensity = 0;
        if (intensitySlider != null) {    currentIntensity = intensitySlider.value;    }
        editLight.GetComponent<Light>().intensity = currentIntensity * 0.01f;
        RenderSettings.ambientIntensity = currentIntensity * 0.01f;
        skyColor = new Color(currentIntensity * 2.55f * (1 / 255), currentIntensity * 2.55f * (1 / 255), currentIntensity * 2.55f * (1 / 255));
        skyMaterial.SetColor("SkyTintColor", skyColor);
    }
}
```

```

public void SetIntensity(float intense) {
    var currentIntensity = intense;
    editLight.GetComponent<Light>().intensity = currentIntensity * 0.01f;
    RenderSettings.ambientIntensity = currentIntensity * 0.01f;
    skyColor = new Color(currentIntensity * 2.55f * (1 / 255), currentIntensity * 2.55f * (1 / 255), currentIntensity * 2.55f * (1 / 255));
    skyMaterial.SetColor("SkyTintColor", skyColor);
    intensitySlider.value = intense;
}
public void SetPositionOfLightSource() {
    posX = float.Parse(posXField.text);
    posZ = float.Parse(posZField.text);
    radius = 20000 * Mathf.Sqrt(2);
    posY = Mathf.Sqrt(radius * radius - Mathf.Pow((posX - 5000), 2) - Mathf.Pow((posZ - 5000), 2));
    editLight.transform.position = new Vector3(posX, posY, posZ);
    editLight.transform.LookAt(new Vector3(5000, 0, 5000));
}
public void SetPositionOfLightSource(float x, float z) {
    posX = x;
    posZ = z;
    radius = 20000 * Mathf.Sqrt(2);
    posY = Mathf.Sqrt(radius * radius - Mathf.Pow((posX - 5000), 2) - Mathf.Pow((posZ - 5000), 2));
    editLight.transform.position = new Vector3(posX, posY, posZ);
    editLight.transform.LookAt(new Vector3(5000, 0, 5000));
}
public void SetLightSource(int type) {
    if (type == 1) {
        moon.SetActive(!moon.activeSelf);
        RenderSettings.skybox = materialMoon;
        SetIntensity(50);
        moon.transform.LookAt(new Vector3(5000, 0, 5000));
    }
    else if (type == 0) {
        SetIntensity(100);
        RenderSettings.skybox = materialSun;
    }
}
}
}

```

«Editor.cs»

```

using System.Collections;
using System.Collections.Generic;
using UnityEngine;
using UnityEngine.EventSystems;
using System;

public class Editor : MonoBehaviour {
    private EditorMode mode = EditorMode.editing;

    private TerrainData tData;

    public GameObject _camera;
    GameObject RightCamera;
    GameObject BottomRightCamera;
    Camera cameraComp;

    public GameObject objectListLoader;
    public GameObject menuClick;
    public GameObject settingsMenu;
    public GameObject waterMenu;

    public GameObject terrainManager;

    public GameObject editingProjector;

    string myName;

    public GameObject editObject;
}

```

```

public bool target = true;
public bool snowSetting = false;

public void SetEditorMode(EditorMode m) {
    mode = m;
    if (m == EditorMode.snow_setting) editingProjector.SetActive(true);
    else editingProjector.SetActive(false);
}

public void SetTerrainData(GameObject t) {
    tData = t.GetComponent<Terrain>().terrainData;
}

void Start () {
    _camera = transform.Find("MainCamera").gameObject;
    RightCamera = transform.Find("RightCamera").gameObject;
    BottomRightCamera = transform.Find("BottomRightCamera").gameObject;
    cameraComp = _camera.GetComponent<Camera>();
}

void Update () {
    Ray ray = cameraComp.ScreenPointToRay(Input.mousePosition);
    RaycastHit rayHit;

    if (Physics.Raycast(ray, out rayHit)) {
        if (mode == EditorMode.snow_setting) {
            editingProjector.transform.position = new Vector3(rayHit.point.x, editingProjector.transform.position.y,
            rayHit.point.z);
        }

        if (Input.GetMouseButtonDown(0) && target) {
            if (EventSystem.current.IsPointerOverGameObject()) {
                return;
            }

            Transform objectHit = rayHit.transform;

            if (mode == EditorMode.editing) {
                if (objectHit.GetComponent<SaveableObj>()) {
                    editObject = objectHit.gameObject;

                    menuClick.GetComponent<onMenuButtonClick>().onMenuClick(4);
                    settingsMenu.GetComponent<settingMenu>().SetEditObject(editObject);
                } else {
                    int mask = 1 << 12;
                    mask = ~mask;

                    Physics.Raycast(rayHit.point + new Vector3(0, 10f, 0), Vector3.down, out rayHit, Mathf.Infinity, mask);
                    objectHit = rayHit.transform;

                    GameObject instObject;
                    myName = objectListLoader.GetComponent<loadObjectList>().GetActiveObjectName();

                    if (!String.IsNullOrEmpty(myName)) {
                        instObject = Instantiate(Resources.Load<GameObject>("EditableObjects/" + myName), rayHit.point,
                        Quaternion.identity);
                        instObject.name = myName;

                        SaveableObj so = instObject.GetComponent<SaveableObj>();

                        if ( (objectHit.GetComponent<Terrain>() && so.swimmingObject) || (objectHit.GetComponent<WaterScript>()
                        && !so.swimmingObject) || (objectHit.GetComponent<IceScript>() && so.swimmingObject) ) {
                            Destroy(instObject);
                            return;
                        }
                    }

                    if (instObject.tag != "YOrient") {
                        instObject.transform.rotation = Quaternion.FromToRotation(Vector3.up, rayHit.normal);
                    }
                }
            }
        }
    }
}

```

```

    }
}
}
else if (mode == EditorMode.snow_setting && !objectHit.GetComponent<WaterScript>()) {
    RaycastHit downRayHit;
    Physics.Raycast(rayHit.point - new Vector3(0, 0.5f, 0), Vector3.up, out downRayHit);

    if (objectHit.GetComponent<Terrain>()) {
        StartCoroutine(terrainManager.GetComponent<SnowTerrainGenerator>().SetSnow(rayHit.point, false));
    }
    else
    if (objectHit.GetComponent<IceScript>()) {
        StartCoroutine(terrainManager.GetComponent<SnowTerrainGenerator>().SetSnow(rayHit.point, true));
    }
    else
    if (downRayHit.transform.GetComponent<IceScript>()) {
        StartCoroutine(terrainManager.GetComponent<SnowTerrainGenerator>().SetSnow(downRayHit.point, true));
    }

    snowSetting = true;
}

}

if (Input.GetMouseButtonDown(1)) {
    Transform objectHit = rayHit.transform;
    if (objectHit.GetComponent<WaterScript>()) {
        menuClick.GetComponent<onMenuButtonClick>().onMenuClick(7);
        waterMenu.GetComponent<waterMenu>().SetActiveWater(objectHit.gameObject);
    }
    else if (objectHit.GetComponent<IceScript>()) {
        menuClick.GetComponent<onMenuButtonClick>().onMenuClick(7);
        GameObject water = objectHit.parent.gameObject;
        waterMenu.GetComponent<waterMenu>().SetActiveWater(water);
    }
}

if (Input.GetMouseButton(0) && snowSetting && mode == EditorMode.snow_setting) {
    if (EventSystem.current.IsPointerOverGameObject()) {
        return;
    }

    Transform objectHit = rayHit.transform;

    RaycastHit downRayHit;
    Physics.Raycast(rayHit.point - new Vector3(0, 1f, 0), Vector3.up, out downRayHit);

    if (objectHit.GetComponent<Terrain>()) {
        StartCoroutine(terrainManager.GetComponent<SnowTerrainGenerator>().SetSnow(rayHit.point, false));
    }
    else
    if (objectHit.GetComponent<IceScript>()) {
        StartCoroutine(terrainManager.GetComponent<SnowTerrainGenerator>().SetSnow(rayHit.point, true));
    }
    else
    if (downRayHit.transform.GetComponent<IceScript>()) {
        StartCoroutine(terrainManager.GetComponent<SnowTerrainGenerator>().SetSnow(downRayHit.point, true));
    }
}
}
}
}
}

```

Федеральное государственное бюджетное образовательное учреждение
высшего образования
«ОРЛОВСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ
ИМЕНИ И.С. ТУРГЕНЕВА»

Институт приборостроения, автоматизации и информационных
технологий

ДНЕВНИК

производственной практики студента

Шорина Владислава Дмитриевича



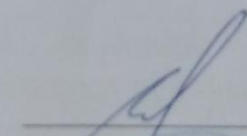
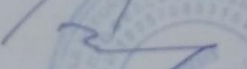
Курс 2

Группа 71ПГ

Место прохождения практики АНО «Центр Интернет-образования»

Руководитель практики
от университета

Руководитель практики
от профильной организации


Чижов А.В.

Забелин С.А.
М.П.

Начало практики
«29» июня 2019 года
Окончание практики
«12» июля 2019 года

График
прохождения практики

№ п/п	Дата	Работа, выполненная студентом	Примечание
1	29.06.2019	Анализ ИЗП на предмет необходимости проведения модернизации	
2	01.07.2019	Анализ ИЗП на предмет необходимости проведения модернизации	
3	02.07.2019	Поиск решений по модернизации найденных проблем в блоке водной поверхности	
4	03.07.2019	Внедрение найденных решений по модернизации в блок водной поверхности	
5	04.07.2019	Отладка внедренных решений в блоке водной поверхности	
6	05.07.2019	Поиск решений по модернизации найденных проблем в блоке снежной поверхности	
7	06.07.2019	Внедрение найденных решений по модернизации в блок снежной поверхности	
8	08.07.2019	Отладка внедренных решений в блоке снежной поверхности	
9	09.07.2019	Поиск решений по модернизации найденных проблем в блоке погодных условий	
10	10.07.2019	Внедрение найденных решений по модернизации в блок погодных условий	
11	11.07.2019	Отладка внедренных решений в блоке погодных условий	
12	12.07.2019	Составление отчета по практике. Защита отчета по практике	

Руководитель практики
от университета

Руководитель практики
от профильной организации

Чижов А.В.

Забелин С.А.

М.П.

Отзыв руководителя практики от профильной организации

Студент Шорин Владислав Дмитриевич

Направление 09.03.04 «Программная инженерия»

Место прохождения практики: АНО «Центр Интернет-образования»

Сроки проведения практики: с «29» июня 2019 г. по «12» июля 2019 г.

Отчет по практике отвечает требованиям выданного индивидуального задания. Все поставленные вопросы разрешены и освещены в достаточной степени.

В ходе прохождения студентом практики наблюдалось ответственное отношение к выполняемой работе, оперативное выполнение поручений, качественный уровень и достаточная степень подготовленности студента к самостоятельному выполнению отдельных заданий.

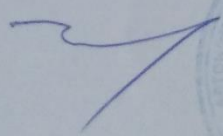
Отчет по практике изложен грамотно, в достаточном объеме, с необходимой для выполнения индивидуального задания степенью использования отечественной и иностранной литературы, а также материалов и справочников.

Дисциплина студента во время практики удовлетворительна. Случаев нарушения трудовой дисциплины не выявлено. Отрицательные черты, действия, проявления, характеризующие студента с негативной стороны в период прохождения практики не обнаружены.

Оценка выполнения программы практики и индивидуального задания в целом: отлично.

Руководитель практики от АНО «Центр Интернет-образования»

Инженер-программист,
Забелин С.А.


«12» 07 2019 г.
М.П.

