**ПРАВИТЕЛЬСТВО РОССИЙСКОЙ ФЕДЕРАЦИИ**

**НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ**

**«ВЫСШАЯ ШКОЛА ЭКОНОМИКИ»**

Факультет компьютерных наук

Департамент программной инженерии

|  |  |  |
| --- | --- | --- |
| СОГЛАСОВАНО  Руководитель департамента программной инженерии факультета компьютерных наук  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ С.М. Авдошин  «\_\_\_» \_\_\_\_\_\_\_\_\_\_\_\_\_ 2019 г. |  | УТВЕРЖДАЮ  Академический руководитель образовательной программы «Программная инженерия» профессор департамента программной инженерии, канд. техн. наук  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ В.В. Шилов  «\_\_\_» \_\_\_\_\_\_\_\_\_\_\_\_\_ 2019 г. |

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  |  | | --- | --- | | ***Подп. и дата*** |  | | ***Инв. № дубл.*** |  | | ***Взам. инв. №*** |  | | ***Подп. и дата*** |  | | ***Инв. № подл*** |  | | **ПРИЛОЖЕНИЕ АКТИВНОСТИ И УВЕДОИЛЕНИЯ ВШЭ**  **Пояснительная записка**  **ЛИСТ УТВЕРЖДЕНИЯ**  **RU.17701729.04.01-01 81 01-1-ЛУ** | | |
|  |  | |
| Исполнитель  студент группы БПИ182  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ /М.А. Шакура /  «\_\_\_\_» \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ 2019 г. | |
|  | | |
|  | |  |

**Москва 2019**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| УТВЕРЖДЕН  RU.17701729.04.01-01 81 01-1-ЛУ |  | |  | |
| |  |  | | --- | --- | | ***Подп. и дата*** |  | | ***Инв. № дубл.*** |  | | ***Взам. инв. №*** |  | | ***Подп. и дата*** |  | | ***Инв. № подл*** |  | | **ПРИЛОЖЕНИЕ АКТИВНОСТИ И УВЕДОИЛЕНИЯ ВШЭ**  **Пояснительная записка**  **RU.17701729.04.01-01 81 01-1**  **Листов 25** | | | | |
|  | |  | | |
|  | | |
|  | | | | |
|  | | | |  |

**Москва 2019**

**АННОТАЦИЯ**

В данном программном документе приведена пояснительная записка к программе «События» («Приложение активности и уведомления ВШЭ»).

В разделе «Введение» указано наименование программы, краткое наименование программы и документы, на основании которых ведется разработка.

В разделе «Назначение и область применения» указано функциональное назначение программы, эксплуатационное назначение программы и краткая характеристика области применения программы.

В разделе «Технические характеристики» содержатся следующие подразделы:

− постановка задачи на разработку программы;

− описание алгоритма и функционирования программы;

− описание метода организации входных и выходных данных;

− описание и обоснование выбора состава технических и программных средств.

В разделе «Ожидаемые технико-экономические показатели» указана предполагаемая потребность и экономические преимущества разработки по сравнению с отечественными и зарубежными образцами или аналогами.

Настоящий документ разработан в соответствии с требованиями:

1) ГОСТ 19.101-77 Виды программ и программных документов;

2) ГОСТ 19.102-77 Стадии разработки;

3) ГОСТ 19.103-77 Обозначения программ и программных документов;

4) ГОСТ 19.104-78 Основные надписи;

5) ГОСТ 19.105-78 Общие требования к программным документам;

6) ГОСТ 19.106-78 Требования к программным документам, выполненным печатным способом;

7) ГОСТ 19.404-79 Пояснительная записка. Требования к содержанию и оформлению.

Изменения к Пояснительной записке оформляются согласно ГОСТ 19.603-78, ГОСТ 19.604-78.

Перед прочтением данного документа рекомендуется ознакомиться с терминологией, приведенной в Приложении 1 настоящей пояснительной записки.

ОГЛАВЛЕНИЕ

[1. ВВЕДЕНИЕ 5](#_Toc8679631)

[1.1. Наименование программы 5](#_Toc8679632)

[1.2. Документы, на основании которых ведется разработка 5](#_Toc8679633)

[2. НАЗНАЧЕНИЕ И ОБЛАСТЬ ПРИМЕНЕНИЯ 6](#_Toc8679634)

[2.1. Назначение программы 6](#_Toc8679635)

[2.1.1. Функциональное назначение 6](#_Toc8679636)

[2.1.2. Эксплуатационное назначение 6](#_Toc8679637)

[2.2. Краткая характеристика области применения 6](#_Toc8679638)

[3. ТЕХНИЧЕСКИЕ ХАРАКТЕРИСТИКИ 7](#_Toc8679639)

[3.1. Постановка задачи на разработку программы 7](#_Toc8679640)

[3.2. Описание алгоритма и функционирования программы 7](#_Toc8679641)

[3.2.1. Описания алгоритма работы взаимодействия сервера и клиента 7](#_Toc8679642)

[3.2.2. Описания алгоритма взаимодействия экранов программы 7](#_Toc8679643)

[3.2.3. Описание работы сервера 8](#_Toc8679644)

[3.2.4. Описание алгоритма загрузки и обновления мероприятий 9](#_Toc8679645)

[3.2.5. Описание алгоритма сохранения данных 10](#_Toc8679646)

[3.2.6. Описание поиска мероприятий 11](#_Toc8679647)

[3.3. Описание и обоснование выбора метода организации входных и выходных данных 11](#_Toc8679648)

[3.3.1. Описание метода организации входных и выходных данных 11](#_Toc8679649)

[3.3.2. Обоснования выбора метода организации входных и выходных данных 12](#_Toc8679650)

[3.4. Описание и обоснование выбора состава технических и программных средств 12](#_Toc8679651)

[3.4.1. Состав технических и программных средств 12](#_Toc8679652)

[3.4.2. Обоснование выбора технических и программных средств 12](#_Toc8679653)

[4. ОЖИДАЕМЫЕ ТЕХНИКО-ЭКОНОМИЧЕСКИЕ ПОКАЗАТЕЛИ 13](#_Toc8679654)

[4.1. Предполагаемая потребность 13](#_Toc8679655)

[4.2. Экономические преимущества разработки по сравнению с отечественными и зарубежными образцами или аналогами 13](#_Toc8679656)

[ПРИЛОЖЕНИЕ 1 14](#_Toc8679657)

[ТЕРМИНОЛОГИЯ 14](#_Toc8679658)

[ПРИЛОЖЕНИЕ 2 15](#_Toc8679659)

[СПИСОК ИСПОЛЬЗОВАННОЙ ЛИТЕРАТУРЫ 15](#_Toc8679660)

[ПРИЛОЖЕНИЕ 3 15](#_Toc8679661)

[АЛГОРИТМ ПОЛЬЗОВАНИЯ 15](#_Toc8679662)

[ПРИЛОЖЕНИЕ 4 16](#_Toc8679663)

[ОПИСАНИЕ И ФУНКЦИОНАЛЬНОЕ НАЗНАЧЕНИЕ КЛАССОВ 16](#_Toc8679664)

[ПРИЛОЖЕНИЕ 5 17](#_Toc8679665)

[ОПИСАНИЕ И ФУНКЦИОНАЛЬНОЕ НАЗНАЧЕНИЕ ПОЛЕЙ, МЕТОДОВ И СВОЙСТВ 17](#_Toc8679666)

# ВВЕДЕНИЕ

## Наименование программы

Наименование программы – «Приложение активности и уведомления ВШЭ» или «HSE Activities and Notifications App».

Наименование программы для пользователя – «События».

## Документы, на основании которых ведется разработка

Приказ декана факультета компьютерных наук Национального исследовательского университета «Высшая школа экономики» №2.3-02/1212-01 от 12.12.2017 «Об утверждении тем, руководителей курсовых работ студентов образовательной программы Программная инженерия факультета компьютерных наук».

# НАЗНАЧЕНИЕ И ОБЛАСТЬ ПРИМЕНЕНИЯ

## Назначение программы

### Функциональное назначение

Функциональным назначением программы является мобильное приложение, помогающее в организации мероприятий НИУ ВШЭ. Программа может как создавать и редактировать мероприятия, так и регистрировать на них участников.

### Эксплуатационное назначение

Функциональным назначением программы является мобильное приложение, помогающее в мониторинге мероприятий НИУ ВШЭ. Программа может искать и добавлять мероприятия в избранное, так же предлагает выбрать язык и тему приложения.

## Краткая характеристика области применения

Приложение активности и уведомления ВШЭ может использоваться как вспомогательный инструмент для мониторинга мероприятий НИУ ВШЭ с использованием мобильного устройства.

# ТЕХНИЧЕСКИЕ ХАРАКТЕРИСТИКИ

## Постановка задачи на разработку программы

Разрабатываемая программа должна соответствовать:

- описанным в техническом задании функциональным характеристикам («Приложение активности и уведомления ВШЭ». Техническое задание)

- описанным в техническом задании требованиям к интерфейсу («Приложение активности и уведомления ВШЭ». Техническое задание)

- описанным в техническом задании требованиям к надежности («Приложение активности и уведомления ВШЭ». Техническое задание)

## Описание алгоритма и функционирования программы

### Описания алгоритма работы взаимодействия сервера и клиента

Программа реализована в виде клиент-серверного приложения.

Клиентское приложение реализовывает следующие функции:

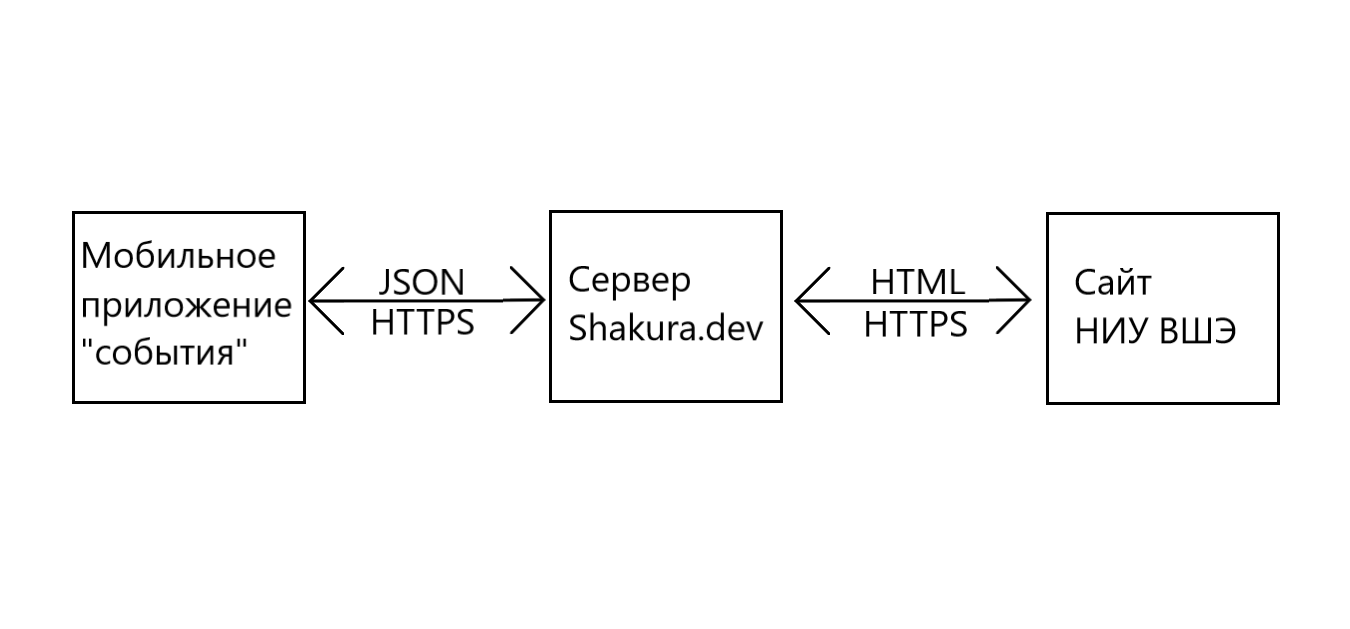
- отправлять запрос серверу на получение информации о мероприятиях;

- предоставляет функционал для работы с мероприятиями;

Серверное приложение должно реализовывать следующие функции:

- раз в час обновлять информацию о мероприятиях в соответствии с сайтом НИУ ВШЭ.

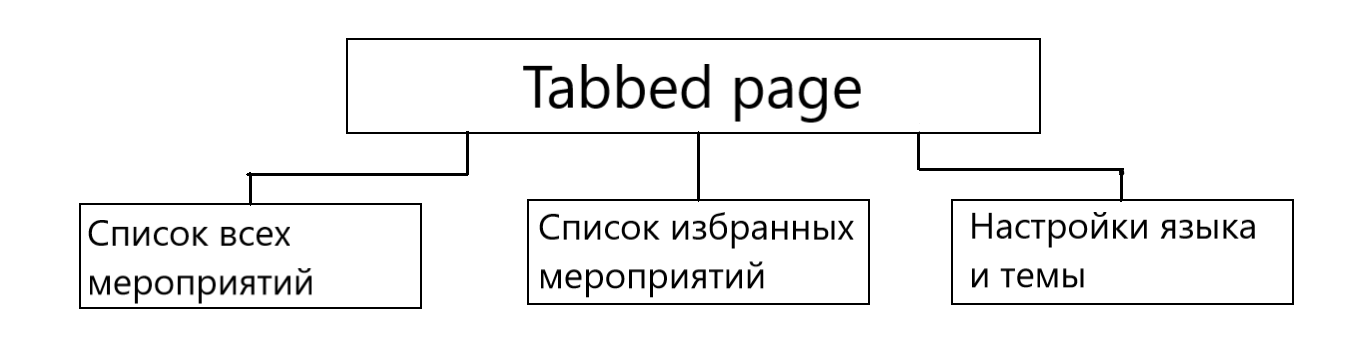
- иметь программный интерфейс приложения для взаимодействия с клиентскими приложениям



*Рисунок 1.* Обмен данными между клиентом, сервером и

### Описания алгоритма взаимодействия экранов программы

Алгоритм взаимодействия экранов программы приведен на рис. 2:



*Рисунок 2.* Взаимодействие экранов программы

### Описание работы сервера

Раз в час сервер делает запрос к сайтам <https://www.hse.ru/news/announcements/> и <https://www.hse.ru/en/news/announcements/>, затем парсит данные о мероприятиях в JSON с помощью Python скрипта:

from flask import json

def hsepars(a):

Events1 = a.findAll("div", {"class": "b-events"})

Announcements = []

for e1 in Events1:

Day = e1.findAll("div", {"class": "b-events\_\_title"})[0].getText()

# Announcements.append({"Type": "Day", "Day": Day})

Events = e1.findAll("div", {"class": "b-events\_\_item js-events-item"})

for e in Events:

Header = e.findAll("div", {"class": "b-events\_\_body\_title large"})[0].getText()

Link = e.findAll("div", {"class": "b-events\_\_body\_title large"})[0].a['href']

PreTags = e.findAll("div", {"class": "tag-set smaller"})[0]

PreExtra = e.findAll("span", {"class": "grey small"})

Time = e.findAll("div", {"class": "b-events\_\_extra date"})[0].getText()

Tags = []

Extra = []

for tag in PreTags:

Tags.append(tag.getText())

for extra in PreExtra:

Extra.append(extra.getText())

Place = ""

try:

Place = e.findAll("p", {"class": "b-text-grey small"})[0].getText()

except:

pass

Announcements.append(

{"Type": "Event", "Header": Header, "Tags": Tags, "Link": Link, "Place": Place, "Extra": Extra,

"Time": Time, "Day": Day})

return json.dumps(Announcements)

### Описание алгоритма загрузки и обновления мероприятий

Обращение к серверу и последующая загрузка мероприятий происходит при запуске приложения и при выполнения жеста обновления списка:

namespace App5.Services

{

public class MockDataStore : IDataStore<Item>

{

List<Item> items;

static public string Get(string url)

{

HttpWebRequest request = (HttpWebRequest)WebRequest.Create(url);

request.AutomaticDecompression = DecompressionMethods.GZip | DecompressionMethods.Deflate;

try

{

using (HttpWebResponse response = (HttpWebResponse)request.GetResponse())

using (Stream stream = response.GetResponseStream())

using (StreamReader reader = new StreamReader(stream))

return reader.ReadToEnd();

}

catch

{

return null;

}

}

static public void RealoadData()

{

AppData.ru = Get("https://shakura.dev/hseapi");

AppData.en = Get("https://shakura.dev/hseapien");

AboutPage.f();

}

public MockDataStore()

{

Item[] data = JsonConvert.DeserializeObject<Item[]>(AppData.isrus ? AppData.ru : AppData.en);

items = data.OfType<Item>().ToList();

}

public async Task<IEnumerable<Item>> GetItemsAsync(bool forceRefresh = false)

{

return await Task.FromResult(items);

}

}

}

### Описание алгоритма сохранения данных

С помощью класса AppData в App5.Models в приложение выполняется сохранение настроек языка и темы, которое происходит после каждого их изменения.

namespace App5.Models

{

struct FileStruct

{

public bool isrus { set; get; }

public bool IsThemeWhite { set; get; }

public List<string> Links;

}

static class AppData

{

static public Action SettingChanged;

static public List<string> Links = new List<string>();

/// <summary>

/// Langs

/// </summary>

static public string ru;

static public string en;

static public bool isrus { set; get; } = true;

/// <summary>

/// Colors

/// </summary>

static public bool IsThemeWhite { set; get; } = true;

static public Color BackgroundColor { get => IsThemeWhite ? Color.White : Color.FromRgb(16, 16, 16); }

static public Color BarBackgroundColor { get => IsThemeWhite ? Color.White : Color.FromRgb(10, 10, 10); }

static public Color FrontColor { get => IsThemeWhite ? Color.Black : Color.White; }

static AppData()

{

SettingChanged += Load;

try

{

FileStruct a;

string json;

string path = Environment.GetFolderPath(Environment.SpecialFolder.Personal);

string filePath = Path.Combine(path, "AppData.txt");

using (var file = File.Open(filePath, FileMode.Open, FileAccess.Read))

using (var strm = new StreamReader(file))

{

json = strm.ReadToEnd();

}

a = JsonConvert.DeserializeObject<FileStruct>(json);

isrus = a.isrus;

IsThemeWhite = a.IsThemeWhite;

Links = a.Links;

}

catch (Exception e){}

}

static void Load()

{

try

{

FileStruct a = new FileStruct() {isrus=isrus,

IsThemeWhite= IsThemeWhite,

Links = Links};

string json = JsonConvert.SerializeObject(a);

string path = Environment.GetFolderPath(Environment.SpecialFolder.Personal);

string filePath = Path.Combine(path, "AppData.txt");

using (var file = File.Open(filePath, FileMode.Create, FileAccess.Write))

using (var strm = new StreamWriter(file))

{

strm.Write(json);

}

}

catch (Exception e){}

}

}

}

### Описание поиска мероприятий

В списке всех мероприятий и избранных мероприятий есть поиск по названию, тегам, месту и времени проведения мероприятий. Список мероприятий изменяется при каждом изменение поля поиска:

async Task ExecuteSearchItems()

{

if (IsBusy)

return;

IsBusy = true;

string SearchQuery = ItemsPage.SearchQuery;

Items.Clear();

string Day = "";

foreach (var item in items)

{

if (item.Header.ToLower().

Contains(SearchQuery.ToLower()) ||

item.Description.ToLower().

Contains(SearchQuery.ToLower()) ||

item.Place.ToLower().

Contains(SearchQuery.ToLower())||

item.Date.ToLower().

Contains(SearchQuery.ToLower()))

{

if (item.Day != Day)

{

Items.Add(new Item() { Type = "Day", Day = Day = item.Day });

}

Items.Add(item);

}

}

IsBusy = false;

}

## Описание и обоснование выбора метода организации входных и выходных данных

### Описание метода организации входных и выходных данных

Входные данные в программе – действия пользователя в приложении (нажатие на кнопку, свайпы). Приложение предоставляет возможность открыть ранее сохранённой и установленной версии в виде apk файла (для смартфона на платформе Android).

Промежуточные входные данные при сборке проекта – данные в формате JSON получаемые от сервера.

Выходные данные в программе – отклики приложения на действия пользователя.

В программе предусмотрена функция сохранения изменений настроек в виде данных типа AppData.

### Обоснования выбора метода организации входных и выходных данных

Механизм десериализации был выбран потому, что он позволяет сократить объем исходного кода и восстановить объекты в их первоначальном виде.

## Описание и обоснование выбора состава технических и программных средств

### Состав технических и программных средств

В состав технических средств должен входить мобильное устройство, включающее в себя:

1) не менее 1 Гб оперативной памяти;

2) свободная память на диске не менее 1 Гб;

3) монитор, разрешение не ниже 800х600 пикселей.

4) Устройства ввода\вывода (сенсорный экран).

Для сервера:

1) Операционная система Linux.

2) не менее 1 Гб оперативной памяти;

3) свободная память на диске не менее 1 Гб;

4) Устройства ввода\вывода.

5) Доступ к сети интернет.

### Обоснование выбора технических и программных средств

Данный состав технических и программных средств позволит программе отображать объекты, осуществлять быстрый отклик на действия пользователя и работать без перебоев.

# ОЖИДАЕМЫЕ ТЕХНИКО-ЭКОНОМИЧЕСКИЕ ПОКАЗАТЕЛИ

## Предполагаемая потребность

Программа для получения информации о мероприятиях. Будет полезна для получения новостей с сайта НИУ ВШЭ;

## Экономические преимущества разработки по сравнению с отечественными и зарубежными образцами или аналогами

В рамках данной работы расчет экономической эффективности не предусмотрен.

# ПРИЛОЖЕНИЕ 1

## ТЕРМИНОЛОГИЯ

Ниже приведен список необходимых терминов для ознакомления.

**Android** - операционная система для смартфонов.

**JSON** - текстовый формат обмена данными, основанный на JavaScript.

**Десериализация** – процесс восстановления данных из формата JSON.

**Сервер –** компьютер с операционной системой, подключенной к сети интеренет.

**Ubuntu –** операционная система для вычислительных машин (в том числе серверов).

**Python –** язык программирования.

**C# –** язык программирования.

**Xamarin –** платформа для разработки мобильных приложений на языке C#.

# ПРИЛОЖЕНИЕ 2

## СПИСОК ИСПОЛЬЗОВАННОЙ ЛИТЕРАТУРЫ

1. Xamarin documentation - <https://docs.microsoft.com/ru-ru/xamarin/>
2. ГОСТ 19.101-77 Виды программ и программных документов. //Единая система программной документации. – М.: ИПК Издательство стандартов, 2001.
3. ГОСТ 19.102-77 Стадии разработки. //Единая система программной документации. – М.: ИПК Издательство стандартов, 2001.
4. ГОСТ 19.103-77 Обозначения программ и программных документов. //Единая система программной документации. – М.: ИПК Издательство стандартов, 2001.
5. ГОСТ 19.104-78 Основные надписи. //Единая система программной документации. – М.: ИПК Издательство стандартов, 2001.
6. ГОСТ 19.105-78 Общие требования к программным документам. //Единая система программной документации. – М.: ИПК Издательство стандартов, 2001.
7. ГОСТ 19.106-78 Требования к программным документам, выполненным печатным способом. //Единая система программной документации. – М.: ИПК Издательство стандартов, 2001.
8. ГОСТ 19.505-79 Руководство оператора. Требования к содержанию и оформлению. //Единая система программной документации. – М.: ИПК Издательство стандартов, 2001.
9. ГОСТ 19.603-78 Общие правила внесения изменений. //Единая система программной документации. – М.: ИПК Издательство стандартов, 2001.
10. ГОСТ 19.604-78 Правила внесения изменений в программные документы, выполненные печатным способом. //Единая система программной документации. – М.: ИПК Издательство стандартов, 2001.

# ПРИЛОЖЕНИЕ 3

## АЛГОРИТМ ПОЛЬЗОВАНИЯ

В нижнем меню пользователь может выбрать один из трех экранов:

1. Список всех мероприятий
2. Список избранных мероприятий
3. Настройки приложения

В первом окне отображается список всех мероприятий. В верхней части экрана доступна кнопка поиска по мероприятиям: по нажатию на нее становится доступа строка ввода, по содержания которой отфильтровываются лишние мероприятия. По клику на мероприятия открывается экран с подробной информацией о мероприятии. Так же появляется возможность открыть страницу мероприятия и добавить его в избранное.

Второе окно аналогично первому, только в списке мероприятий находятся только избранные.

В третьем окне находится список настроек: цветовой и языковой тем. Для удобства пользователя, каждое изменение настроек фиксируется в память телефона и автоматически применяется при следующем запуске.

# ПРИЛОЖЕНИЕ 4

## ОПИСАНИЕ И ФУНКЦИОНАЛЬНОЕ НАЗНАЧЕНИЕ КЛАССОВ

Таблица 4.1

Описание и функциональное назначение классов

|  |  |
| --- | --- |
| Класс | Назначение |
| AppData | Фиксирует все данные в программе. |
| Item | Является классом хранения данных мероприятия. |
| MockDataStore | Обновляет данные присылаемые сервером. |
| AboutViewModel | Модель для меню настроек. |
| BaseViewModel | Класс от которого наследуется все модели экранов. |
| ItemDetailViewModel | Модель экрана подробностей мероприятия. |
| ItemsViewModel | Модель списков мероприятий. |
| AboutPage | Контролирует экран настроек. |
| Favorite | Контролирует экран избранных мероприятий. |
| ItemDetailPage | Контролирует экран подробностей мероприятий. |
| ItemsPage | Контролирует экран списка всех мероприятий. |
| MainPage | Контролирует общий экран приложения. |
| App | Контролирует меню уровней. |

# ПРИЛОЖЕНИЕ 5

## ОПИСАНИЕ И ФУНКЦИОНАЛЬНОЕ НАЗНАЧЕНИЕ ПОЛЕЙ, МЕТОДОВ И СВОЙСТВ

Таблица 5.1

Описание полей, методов и свойств класса AppData.cs

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Поля** | |  | |  | |  | |
| Имя | | Модификатор доступа | | Тип | | Назначение | |
| ru | | public | | string | | Список русскоязычных мероприятий | |
| en | | public | | string | | Список англоязычных мероприятий | |
| isrus | | public | | bool | | Является ли выбранный язык русским | |
| IsThemeWhite | | public | | bool | | Является ли выбранная тема светлой | |
| BackgroundColor | | public | | Color | | Цвет фона. Возвращается в зависимости от IsThemeWhite. | |
| BarBackgroundColor | | public | | Color | | Цвет фона NavigationBar. Возвращается в зависимости от IsThemeWhite. | |
| FrontColor | | public | | Color | | Цвет текста. Возвращается в зависимости от IsThemeWhite. | |
| Links | | public | | void | | Список ссылок на избранные мероприятия | |
| SettingChanged | | public | | Action | | Событие изменения настроек. Вызывает Laod(). | |
| **Методы** | |  | |  | |  | |
| Имя | Модификатор доступа | | Тип | | Аргументы | | Назначение |
| AppData | public | | void | | - | | Выгружает данные из памяти. |
| Load | public | | void | | - | | Загружает данные в память |

Таблица 5.2

Описание полей, методов и свойств класса Item.cs

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Поля** | |  | |  | |  | |
| Имя | | Модификатор доступа | | Тип | | Назначение | |
| Type | | public | | string | | Тип ячейки. Принимает значения “Day ” и ”Event” | |
| Day | | public | | string | | День мероприятия | |
| Header | | public | | string | | Заголовок мероприятия | |
| Description | | public | | string | | Доп. информация мероприятия | |
| ExtraString | | public | | string | | Тэги мероприятия | |
| Link | | public | | string | | Ссылка на мероприятие | |
| Place | | public | | string | | Место проведения мероприятия | |
| Time | | public | | string | | Время проведения мероприятия | |
| Date | | public | | string | | Дата и время проведения мероприятия | |
| Tags | | public | | string[] | | Маcсив тэгов мероприятия | |
| Extra | | public | | string[] | | Массив строк доп. информации о мероприятии | |
| DayVisible | | public | | bool | | Должен ли отображаться день мероприятия | |
| ExtraVisible | | public | | bool | | Должна ли отображаться доп. информация мероприятия | |
| DescriptionVisible | | public | | bool | | Должны ли отображаться тэги мероприятия | |
| TimeVisible | | public | | bool | | Должно ли отображаться время мероприятия | |
| PlaceVisible | | public | | bool | | Должно ли отображаться место мероприятия | |
| TagsVisible | | public | | bool | | Должны ли отображаться тэги мероприятия | |
| HeaderVisible | | public | | bool | | Должен ли отображаться заголовок мероприятия | |
| Clickable | | public | | bool | | Можно ли открыть данную ячейку. Зависит от поля Type. | |
| FavSource | | public | | string | | Иконка сохранения мероприятия. Зависит от того, было ли оно уже добавлено в избранное. | |
| FrontColor | | public | | Color | | Цвет текста. Зависит от цветовой темы. | |
| BackgroundColor | | public | | Color | | Цвет фона. Зависит от цветовой темы. | |
| **Методы** | |  | |  | |  | |
| Имя | Модификатор доступа | | Тип | | Аргументы | | Назначение |

Таблица 5.3

Описание полей, методов и свойств класса MockDataStore.cs

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Поля** | |  | |  | |  |
| Имя | | Модификатор доступа | | Тип | | Назначение |
| items | | public | | List<Item> | | Список всех мероприятий |
| **Методы** | |  | |  | |  |
| Имя | Модификатор доступа | | Тип | | Аргументы | Назначение |
| Items | public | | string | | string url | Отправляет запрос на сервер. |
| RealoadData | public | | void | | - | Отправляет запрос к серверу, обновляет данные. |
| MockDataStore | public | | void | | - | Обновляет данные при изменении языка. |

Таблица 5.4

Описание полей, методов и свойств класса ItemsViewModel.cs

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Поля** | |  | |  | | |  |
| Имя | | Модификатор доступа | | Тип | | | Назначение |
| Items | | public | | ObservableCollection<Item> | | | Список мероприятий |
| FavItems | | public | | ObservableCollection<Item> | | | Список избранных |
| items | | public | | System.Collections.Generic.IEnumerable<Item> | | | Список всех мероприятий |
| LoadItemsCommand | | public | | Command | | | Обновление мероприятий |
| LoadItemsCommand1 | | public | | Command | | | Загрузка мероприятий |
| SearchItems | | public | | Command | | | Обновление поиска мероприятий |
| **Методы** | |  | |  | | |  |
| Имя | Модификатор доступа | | Тип | | Аргументы | Назначение | |
| ItemsViewModel | public | | void | | - | Конструктор класса. | |
| ExecuteLoadItemsCommand | private | | async Task | | - | Загрузка списка событий. | |
| ExecuteSearchItems | private | | async Task | | - | Поиск по всем событиям. | |
| ExecuteSearchFavotitesItems | private | | async Task | | - | Поиск по избранным событиям. | |

Таблица 5.5

Описание полей, методов и свойств класса AboutPage.cs

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Поля** | |  | |  | |  | |
| Имя | | Модификатор доступа | | Тип | | Назначение | |
| ThemeChanged | | public | | event Action | | Вызывается при изменении темы. | |
| **Методы** | |  | |  | |  | |
| Имя | Модификатор доступа | | Тип | | Аргументы | | Назначение |
| AboutPage | public | | void | | - | | Вызывается в начале работы скрипта. |
| Russian\_Clicked | private | | async void | | object sender, EventArgs args | | Нажатие на кнопку русского языка. |
| English\_Clicked | private | | async void | | object sender, EventArgs args | | Нажатие на кнопку английского языка. |
| Black\_Clicked | private | | async void | | object sender, EventArgs args | | Нажатие на кнопку темной темы. |
| White\_Clicked | private | | async void | | object sender, EventArgs args | | Нажатие на кнопку светлой темы. |
| UIUpdate | private | | async void | | int id | | Обновляет внешний вид программы. |

Таблица 5.6

Описание полей, методов и свойств класса Favorite.xaml.cs

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Поля** | |  | | |  | | | |  | |
| Имя | | Модификатор доступа | | | Тип | | | | Назначение | |
| viewModel | | private | | | ItemsViewModel | | | | Макет модели. | |
| SearchQuery | | public | | | string | | | | Поисковый запрос по мероприятиям | |
| **Методы** | | |  | | |  | |  | | |
| Имя | Модификатор доступа | | | Тип | | | Аргументы | | | Назначение |
| SearchQueryUpdate | private | | | void | | | object sender, EventArgs e | | | Обновление строки поиска |
| Favorite | public | | | void | | | - | | | конструктор |
| OnItemSelected | private | | | void | | | object sender, EventArgs e | | | Обработчик нажатия на мероприятие |
| Search\_Clicked | private | | | void | | | object sender, EventArgs e | | | Обработчик нажатия на поиск |
| UIUpdate | private | | | void | | | object sender, EventArgs e | | | Обновляет внешний вид окна |

Таблица 5.7

Описание полей, методов и свойств класса ItemsPage.xaml.cs

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Поля** | |  | |  | |  | |
| Имя | | Модификатор доступа | | Тип | | Назначение | |
| viewModel | | private | | ItemsViewModel | | Макет модели. | |
| SearchQuery | | public | | string | | Поисковый запрос по мероприятиям | |
| **Методы** | |  | |  | |  | |
| Имя | Модификатор доступа | | Тип | | Аргументы | | Назначение |
| SearchQueryUpdate | private | | void | | object sender, EventArgs e | | Обновление строки поиска |
| Favorite | public | | void | | - | | конструктор |
| OnItemSelected | private | | void | | object sender, EventArgs e | | Обработчик нажатия на мероприятие |
| Search\_Clicked | private | | void | | object sender, EventArgs e | | Обработчик нажатия на поиск |
| UIUpdate | private | | void | | object sender, EventArgs e | | Обновляет внешний вид окна |
| SearchQueryUpdate | private | | void | | object sender, EventArgs e | | Обновление строки поиска |

Таблица 5.8

Описание полей, методов и свойств класса ItemDetailPage.xaml.cs

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Поля** | |  | |  | |  | |
| Имя | | Модификатор доступа | | Тип | | Назначение | |
| viewModel | | private | | ItemDetailViewModel | | Макет модели. | |
| **Методы** | |  | |  | |  | |
| Имя | Модификатор доступа | | Тип | | Аргументы | | Назначение |
| ItemDetailPage | private | | void | | ItemDetailViewModel viewModel | | Конструктор. |
| OpenLink | private | | void | | - | | Открывает ссылку на мероприятие |
| FavClicked | private | | void | | object sender, EventArgs e | | Добавляет/Убирает в/из избранного. |
| OnButtonClicked | private | | void | | object sender, EventArgs e | | Обработчик нажатия кнопки перехода на вебсайт. |
| UIUpdate | private | | void | | - | | Обновление интерфейса. |

Таблица 5.9

Описание полей, методов и свойств класса MainPage.xaml.cs

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Методы** | |  | |  | |  | |
| Имя | Модификатор доступа | | Тип | | Аргументы | | Назначение |
| MainPage | public | | void | | - | | Конструктор. |
| UIUpdate | private | | void | | - | | Обновление интерфейса. |

Таблица 5.10

Описание полей, методов и свойств класса LevelSave.cs

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Поля** | |  | |  | |  | |
| Имя | | Модификатор доступа | | Тип | | Назначение | |
| planets | | public | | List<PlayObject> | | Список планет. | |
| name | | public | | string | | Название уровня. | |
| **Методы** | |  | |  | |  | |
| Имя | Модификатор доступа | | Тип | | Аргументы | | Назначение |
| SaveObj | public | | void | | string key | | Сохранение объекта. |
| LevelSave | public | | конструктор | | - | | Пустой конструктор. |
| LevelSave | public | | конструктор | | string key | | Конструктор с парметром. |

Таблица 5.11

Описание полей, методов и свойств класса App.cs

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Методы** | |  | |  | |  | |
| Имя | Модификатор доступа | | Тип | | Аргументы | | Назначение |
| App | public | | void | | - | | Конструктор. |
| OnStart | public | | void | | - | | Функция, вызываемая на старте. |
| OnSleep | public | | void | | - | | Функция, вызываемая при остановке. |
| OnResume | public | | void | | - | | Функция, вызываемая при возвращении. |

Таблица 5.12

ЛИСТ РЕГИСТРАЦИИ ИЗМЕНЕНИЙ

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Лист регистрации изменений | | | | | | | | | |
| Номера листов (страниц) | | | | | Всего листов (страниц в докум.) | № документа | Входящий № сопроводительного докум. и дата | Подп. | Дата |
| Изм. | Измененных | Замененных | Новых | Аннулированных |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |