АВТОНОМНАЯ НЕКОММЕРЧЕСКАЯ ОРГАНИЗАЦИЯ
ВЫСШЕГО ОБРАЗОВАНИЯ
«СЕВЕРО-КАВКАЗСКИЙ СОЦИАЛЬНЫЙ ИНСТИТУТ»

Утверждаю
Декан факультета
_____Ж.В. Игнатенко
«28» 10 2020г.

# РАБОЧАЯ ПРОГРАММА ДИСЦИПЛИНЫ

Разработка мобильных приложений

Направление подготовки 09.03.03 Прикладная информатика

Направленность (профиль) программы Прикладная информатика в экономике

Квалификация выпускника бакалавр

Форма обучения очная, заочная

год начала подготовки – 2019

Разработана
канд.техн.наук, доцент, доцент
_____О.Х. Шаяхметов

Согласована
зав. выпускающей кафедры
_____Ж.В. Игнатенко

Рекомендована
на заседании кафедры
от «28» 10 2020г.
протокол № 2
Зав. кафедрой _____ Ж.В. Игнатенко

Одобрена
на заседании учебно-методической
комиссии факультета
от «28» 10 2020г.
протокол № 2
Председатель УМК _____ Ж.В.
Игнатенко

Ставрополь, 2020 г.

**Содержание**

1. Цели освоения дисциплины ....3
2. Место дисциплины в структуре ОПОП....3
3. Планируемые результаты обучения по дисциплине....3
4. Объем дисциплины и виды учебной работы ....4
5. Содержание и структура дисциплины....5
5.1. Содержание дисциплины ....5
5.2. Структура дисциплины....6
5.3. Занятия семинарского типа....7
5.4. Курсовой проект (курсовая работа, расчетно-графическая работа, реферат, контрольная работа)....8
5.5. Самостоятельная работа....8
6. Образовательные технологии....8
6. Образовательные технологии....8
7. Фонд оценочных средств (оценочные материалы) для текущего контроля успеваемости, промежуточной аттестации....9
8. Учебно-методическое и информационное обеспечение дисциплины....9
8.1. Основная литература ....9
8.2. Дополнительная литература....9
8.3. Программное обеспечение ....9
8.4. Информационно-справочные системы ....10
8.5. Информационные справочные системы ....10
8.6. Интернет-ресурсы ....10
8.7. Методические указания по освоению дисциплины....10
9. Материально-техническое обеспечение дисциплины ....12
10. Особенности освоения дисциплины лицами с ограниченными возможностями здоровья ....12
ППриложение 1....14

## 1. ЦЕЛИ ОСВОЕНИЯ ДИСЦИПЛИНЫ

Цель освоения дисциплины- изучение базового устройства популярных мобильных платформ и возможностей, которые предоставляет данная платформа для разработки мобильных систем на базе эмуляторов, получение практических навыков по созданию пользовательских интерфейсов, сервисов, а также по использованию сигнализации, аппаратных сенсоров и стандартных хранилищ информации популярных мобильных платформ. В указанном курсе обучаемые должны приобрести устойчивые знания по программированию мобильных гаджетов, сервисов, служб.

## 2. МЕСТО ДИСЦИПЛИНЫ В СТРУКТУРЕ ОПОП

Дисциплина (Б.1.В.6) «Разработка мобильных приложений» входит в часть, формируемую участниками образовательных отношений, – обязательные дисциплины Блока 1 «Дисциплины (модули)» и находится в логической и содержательно-методической связи с другими дисциплинами.

| Предшествующие дисциплины (курсы, модули, практики) | Последующие дисциплины (курсы, модули, практики) |
|---|---|
| Информатика и программирование<br>Дискретная математика<br>Программная инженерия<br>Разработка и стандартизация программных средств и информационных технологий<br>Информационная безопасность<br>Разработка программных приложений<br>Облачные технологии | Интернет-программирование<br>Визуальное программирование<br>Производственная (технологическая (проектно-технологическая)практика)практика<br>Производственная практика (преддипломная практика) |

Требования к «входным» знаниям, умениям и навыкам обучающегося, необходимым при освоении данной дисциплины:

Знать: понятие, основные свойства и этапы разработки алгоритмов, способы и формы их представления; основные типы алгоритмических структур, понятие вычислительного процесса и его взаимосвязь с понятием алгоритма; основные этапы решения задач с использованием ЭВМ, структуру и возможности систем программирования, методы и этапы разработки программных продуктов; понятие языка программирования как системы обозначений для описания алгоритма, классификацию языков программирования и основные направления их развития, структуру алгоритмических языков, понятия синтаксиса и семантики языка, формы описания синтаксических конструкций; концепцию типов данных в языках программирования высокого уровня, базовые и производные– типы данных, набор функций и операций, допустимых для каждого из них, правила приведения типов в выражениях; основные идеи, принципы и методы структурного программирования.

Уметь: применять правила записи алгоритмов и программ, базовые управляющие структуры: последовательность, ветвление, цикл и их реализацию.

Владеть: навыками обработки информации с применением информационно-коммуникационных технологий и с учетом основных требований информационной безопасности

## 3. ПЛАНИРУЕМЫЕ РЕЗУЛЬТАТЫ ОБУЧЕНИЯ ПО ДИСЦИПЛИНЕ

| Формируемые компетенции (код компетенции, наименование) | Планируемые результаты обучения |
|---|---|
| ПК-2 Способность разрабатывать и адаптировать прикладное программное обеспечение | Знать: современные объектно-ориентированные языки программирования; современные структурные языки программирования. |

| | |
|---|---|
| | Уметь: кодировать на языках программирования; тестировать результаты прототипирования; верифицировать структуру программного кода.<br>Владеть: навыками разработки структуры программного кода; верификации структуры программного кода относительно архитектуры ИС и требований заказчика к ИС; разработки руководства программиста ИС; обеспечения соответствия разработанного кода и процесса кодирования на языках программирования принятым в организации или проекте стандартам и технологиям. |
| ПК-8 Способность проводить тестирование компонентов программного обеспечения ИС | Знать:инструменты и методы модульного тестирования; регламенты модульного тестирования.<br>Владеть: навыками обеспечения соответствия процессов модульного тестирования ИС принятым в организации или проекте стандартам и технологиям. |

## 4. ОБЪЕМ ДИСЦИПЛИНЫ И ВИДЫ УЧЕБНОЙ РАБОТЫ

Общий объем дисциплины составляет 5 зачетных единиц, 180 академических часа.
Очная форма обучения

| Вид учебной работы | Всего часов | Триместры | | |
|---|---|---|---|---|
| | | 9 | | |
| **Контактная работа (всего)** | **38,5** | 38,5 | | |
| в том числе: | | | | |
| 1) занятия лекционного типа (ЛК) | 18 | 18 | | |
| из них | | | | |
| – лекции | | | | |
| 2) занятия семинарского типа (ПЗ) | 18 | 18 | | |
| из них | | | | |
| – семинары (С) | | | | |
| – практические занятия (ПР) | | | | |
| – лабораторные работы (ЛР) | 18 | 18 | | |
| 3) групповые консультации | 2 | 2 | | |
| 4) индивидуальная работа | | | | |
| 5) промежуточная аттестация | 0,5 | 0,5 | | |
| **Самостоятельная работа (всего) (СР)** | **141,5** | 141,5 | | |
| в том числе: | | | | |
| Курсовой проект (работа) | | | | |
| Расчетно-графические работы | | | | |
| Контрольная работа | | | | |
| Реферат | 20 | 20 | | |
| Самоподготовка (самостоятельное изучение разделов, проработка и повторение лекционного материала и материала учебников и учебных пособий, подготовка к лабораторным и практическим занятиям, коллоквиумами т.д.) | 95 | 95 | | |

| | | | | |
|---|---|---|---|---|
| Подготовка к аттестации | 26,5 | 26,5 | | |
| Общий объем, час | 180 | 180 | | |
| Форма промежуточной аттестации(Экзамен) | экзамен | экзамен | | |

Заочная форма обучения

| Вид учебной работы | Всего часов | Триместры | | |
|---|---|---|---|---|
| | | А | | |
| **Контактная работа (всего)** | **20,5** | 20,5 | | |
| в том числе: | | | | |
| 1) занятия лекционного типа (ЛК) | 8 | 8 | | |
| из них | | | | |
| – лекции | | | | |
| 2) занятия семинарского типа (ПЗ) | 12 | 12 | | |
| из них | | | | |
| – семинары (С) | | | | |
| – практические занятия (ПР) | | | | |
| – лабораторные работы (ЛР) | 12 | 12 | | |
| 3) групповые консультации | | | | |
| 4) индивидуальная работа | | | | |
| 5) промежуточная аттестация | 0,5 | 0,5 | | |
| **Самостоятельная работа (всего) (СР)** | **159,5** | 159,5 | | |
| в том числе: | | | | |
| Курсовой проект (работа) | | | | |
| Расчетно-графические работы | | | | |
| Контрольная работа | | | | |
| Реферат | - | - | | |
| Самоподготовка (самостоятельное изучение разделов, проработка и повторение лекционного материала и материала учебников и учебных пособий, подготовка к лабораторным и практическим занятиям, коллоквиумами т.д.) | 151 | 151 | | |
| Подготовка к аттестации | 8,5 | 8,5 | | |
| Общий объем, час | 180 | 180 | | |
| Форма промежуточной аттестации (экзамен) | экзамен | экзамен | | |

## 5. СОДЕРЖАНИЕ И СТРУКТУРА ДИСЦИПЛИНЫ

### 5.1. Содержание дисциплины

| № раздела (темы) | Наименование раздела (темы) | Содержание раздела (темы) |
|---|---|---|
| 1 | Введение в мобильное программирование. Операционная система Windows Phone 7 | Мобильное программирование, платформы для разработки. Система Windows Phone 7. Microsoft Visual Studio Express for Windows Phone. Аппаратные средства устройств, поддерживающих Windows Phone 7. |
| 2 | Введение в Silverlight | Проектирование программы Silverlight. Язык XAML. Пример создания приложения Silverlight для Windows Phone. |

| | | |
|---|---|---|
| 3 | Управление решениями в Visual Studio | Управление решениями в Visual Studio. Проекты и решения в Visual Studio. Отладка программ. |
| 4 | Создание приложений Silverlight | Улучшение приложения. Изменение и отображение данных. Управление ориентацией страницы приложения. Отображение списков данных. Навигация по страницам приложения. Использование классов ViewModel. |
| 5 | Хранение данных приложений | Хранилище данных Windows Phone. Базы данных в Windows Phone. Создание связей данных в LINQ. |
| 6 | Средства Windows Phone для работы с сетью | Основные сведения о сетях. Создание подключения по протоколу UDP. Создание подключения по протоколу TCP. Подключение к сетевому ресурсу. Чтение данных из XML-потока с помощью LINQ. Взаимодействие приложений с сетевыми службами. |
| 7 | Создание приложений XNA | Основные сведения о технологии XNA. Создание игрового мира. Использование средств Windows Phone в играх. Совместное использование XNA и Silverlight. |
| 8 | Использование системных функций в приложениях | Значки и экраны-заставки приложений для Windows Phone. Быстрое переключение приложений. Задачи запуска и задачи выбора. Фоновые задачи. |
| 9 | Публикация приложений в Windows Phone Marketplace | Подготовка приложения для продажи. Распространение приложений и игр для Windows Phone. |

**5.2. Структура дисциплины**

Очная форма обучения

| № раздела (темы) | Наименование раздела (темы) | Количество часов | | | | | |
|---|---|---|---|---|---|---|---|
| | | Всего | ЛК | ЛР | ПР | С | СР |
| 1 | Введение в мобильное программирование. Операционная система Windows Phone 7 | 16 | 2 | 2 | | | 12 |
| 2 | Введение в Silverlight | 16 | 2 | 2 | | | 12 |
| 3 | Управление решениями в Visual Studio | 16 | 2 | 2 | | | 12 |
| 4 | Создание приложений Silverlight | 16 | 2 | 2 | | | 12 |
| 5 | Хранение данных приложений | 16 | 2 | 2 | | | 12 |
| 6 | Средства Windows Phone для работы с сетью | 16 | 2 | 2 | | | 12 |
| 7 | Создание приложений XNA | 16 | 2 | 2 | | | 12 |
| 8 | Использование системных функций в приложениях | 16 | 2 | 2 | | | 12 |
| 9 | Публикация приложений в Windows Phone Marketplace | 23 | 2 | 2 | | | 19 |
| | Групповая консультация | 2 | | | | | |
| | Промежуточная аттестация | 27 | | | | | |
| | Общий объем | 180 | 18 | 18 | | | 115 |

Заочная форма обучения

| № раздела (темы) | Наименование раздела (темы) | Количество часов | | | | | |
|---|---|---|---|---|---|---|---|
| | | Всего | ЛК | ЛР | ПР | С | СР |
| 1 | Введение в мобильное программирование. Операционная система Windows Phone 7 | 18 | 1 | 1 | | | 16 |
| 2 | Введение в Silverlight | 18 | 1 | 1 | | | 16 |
| 3 | Управление решениями в Visual Studio | 19 | 1 | 2 | | | 16 |
| 4 | Создание приложений Silverlight | 19 | 1 | 2 | | | 16 |
| 5 | Хранение данных приложений | 19 | 1 | 2 | | | 16 |
| 6 | Средства Windows Phone для работы с сетью | 19 | 1 | 2 | | | 16 |
| 7 | Создание приложений XNA | 19 | 1 | 2 | | | 16 |
| 8 | Использование системных функций в приложениях | 17 | 1 | | | | 16 |
| 9 | Публикация приложений в Windows Phone Marketplace | 23 | | | | | 23 |
| | Групповая консультация | - | | | | | |
| | Промежуточная аттестация | 9 | | | | | |
| | Общий объем | 180 | 8 | 12 | | | 151 |

**5.3. Занятия семинарского типа**

очная форма обучения

| № п/п | № раздела (темы) | Вид занятия | Наименование | Количество часов |
|---|---|---|---|---|
| 1 | 1 | ЛР | Пакет Windows Phone SDK иэмуляторWindows Phone | 2 |
| 2 | 2 | ЛР | Создание простого приложенияSilverlight | 2 |
| 3 | 3 | ЛР | Отладкаприложения | 2 |
| 4 | 4 | ЛР | Использование нескольких проектов в одном решении | 2 |
| 5 | 5 | ЛР | Обработка ошибок вводаданных | 2 |
| 6 | 6 | ЛР | Привязкаданных | 2 |
| 7 | 7 | ЛР | Использование различных режимовориентации | 2 |
| 8 | 8 | ЛР | Отображение списковданных | 2 |
| 9 | 9 | ЛР | Многостраничныеприложения | 2 |

заочная форма обучения

| № п/п | № раздела (темы) | Вид занятия | Наименование | Количество часов |
|---|---|---|---|---|
| 1 | 1 | ЛР | Пакет Windows Phone SDK иэмуляторWindows Phone | 1 |
| 2 | 2 | ЛР | Создание простого приложенияSilverlight | 1 |
| 3 | 3 | ЛР | Отладкаприложения | 2 |
| 4 | 4 | ЛР | Использование нескольких проектов в одном решении | 2 |
| 5 | 5 | ЛР | Обработка ошибок вводаданных | 2 |

| | | | | |
|---|---|---|---|---|
| 6 | 6 | ЛР | Привязкаданных | 2 |
| 7 | 7 | ЛР | Использование различных режимовориентации | 2 |
| 8 | 8 | ЛР | Отображение списковданных | |
| 9 | 9 | ЛР | Многостраничныеприложения | |

**5.4. Курсовой проект (курсовая работа, расчетно-графическая работа, реферат, контрольная работа)**

Не предусмотрено

**5.5. Самостоятельная работа**

очная форма обучения

| № раздела (темы) | Виды самостоятельной работы | Количество часов |
|---|---|---|
| 1-9 | Проработка и повторение лекционного материала | 45 |
| 1-9 | Подготовка к практическим занятиям | 50 |
| 1-9 | Подготовка к практическим занятиям | 20 |
| | Подготовка к аттестации | 26,5 |
| | Итого: | 141,5 |

заочная форма обучения

| № раздела (темы) | Виды самостоятельной работы | Количество часов |
|---|---|---|
| 1-9 | Проработка и повторение лекционного материала | 65 |
| 1-9 | Подготовка к практическим занятиям | 66 |
| 1-9 | Подготовка к практическим занятиям | 20 |
| | Подготовка к аттестации | 8,5 |
| | Итого: | 159,5 |

**6. ОБРАЗОВАТЕЛЬНЫЕ ТЕХНОЛОГИИ**

**6. ОБРАЗОВАТЕЛЬНЫЕ ТЕХНОЛОГИИ**

*Информационные технологии, используемые при осуществлении образовательного процесса по дисциплине:*

– сбор, хранение, систематизация, обработка и представление учебной и научной информации;

– обработка различного рода информации с применением современных информационных технологий;

– самостоятельный поиск дополнительного учебного и научного материала, с использованием поисковых систем и сайтов сети Интернет, электронных энциклопедий и баз данных;

– использование электронной почты для рассылки и асинхронного общения, чата преподавателей и обучающихся, переписки и обсуждения возникших учебных проблем для синхронного взаимодействия

дистанционные образовательные технологии (при необходимости).

Практическая подготовка обучающихся не предусмотрена

**Интерактивные и активные образовательные технологии**

| № раздела | Вид занятия (ЛК, ПР, С, | Используемые интерактивные и активные образовательные технологии | Количество часов |
|---|---|---|---|

| (темы) | ЛР) | | ОФО/ЗФО |
|---|---|---|---|
| 1 | Л | Виртуальная экскурсия «Операционная система WindowsPhone 7». | 1/1 |
| 2 | Л | Виртуальная экскурсия «Методы криптографии». | 1/1 |
| 3 | Л | Дискуссия. | 1/1 |
| 4 | ПЗ | Опережающая самостоятельная работа студентов. | 1/1 |

Практическая подготовка обучающихся не предусмотрена

## 7. ФОНД ОЦЕНОЧНЫХ СРЕДСТВ (ОЦЕНОЧНЫЕ МАТЕРИАЛЫ) ДЛЯ ТЕКУЩЕГО КОНТРОЛЯ УСПЕВАЕМОСТИ, ПРОМЕЖУТОЧНОЙ АТТЕСТАЦИИ

Фонд оценочных средств(оценочные материалы) для текущего контроля успеваемости, промежуточной аттестации по дисциплине приводятся в приложении.

## 8. УЧЕБНО-МЕТОДИЧЕСКОЕ И ИНФОРМАЦИОННОЕ ОБЕСПЕЧЕНИЕ ДИСЦИПЛИНЫ

### 8.1. Основная литература

1. Соколова, В. В. Вычислительная техника и информационные технологии. Разработка мобильных приложений : учебное пособие для вузов / В. В. Соколова. — Москва : Издательство Юрайт, 2020. — 175 с. — (Высшее образование). — ISBN 978-5-9916-6525-4. — Текст : электронный // ЭБС Юрайт [сайт]. — URL: https://urait.ru/bcode/451366.

2.Введение в разработку приложений для ОС Android [Электронный ресурс]: учебное пособие/ Ю.В. Березовская [и др.].— Электрон. текстовые данные.— Москва: Интернет-Университет Информационных Технологий (ИНТУИТ), Ай Пи Ар Медиа, 2021.— 427 с.— Режим доступа: http://www.iprbookshop.ru/102000.html.— ЭБС «IPRbooks»

3.Семакова А. Введение в разработку приложений для смартфонов на ОС Android [Электронный ресурс]: учебное пособие/ Семакова А.— Электрон. текстовые данные.— Москва: Интернет-Университет Информационных Технологий (ИНТУИТ), Ай Пи Ар Медиа, 2021.— 102 с.— Режим доступа: http://www.iprbookshop.ru/102001.html.— ЭБС «IPRbooks»

### 8.2. Дополнительная литература

1. Соколова, В. В. Разработка мобильных приложений : учебное пособие для среднего профессионального образования / В. В. Соколова. — Москва : Издательство Юрайт, 2020. — 175 с. — (Профессиональное образование). — ISBN 978-5-534-10680-0. — Текст : электронный // ЭБС Юрайт [сайт]. — URL: https://urait.ru/bcode/456795.

2. Кокорева Е.В. Технология разработки телекоммуникационных сервисов. Распределённые приложения [Электронный ресурс]: учебно-методическое пособие/ Кокорева Е.В.— Электрон. текстовые данные.— Новосибирск: Сибирский государственный университет телекоммуникаций и информатики, 2019.— 104 с.— Режим доступа: http://www.iprbookshop.ru/102142.html.— ЭБС «IPRbooks»

### 8.3. Программное обеспечение

1. Операционная система семейства WindowsMicrosoft
2. MSOffice
3. Visual C#

4. Visual Studio

### 8.4. Информационно-справочные системы

Консультант Плюс

### 8.5. Информационные справочные системы

Разработка приложений для WindowsPhone 7 [Электронный ресурс]. - http://msdn.microsoft.com/ru-ru/windowsphone/default.aspx

### 8.6. Интернет-ресурсы

1. Интернет университет информационных технологий [Электронный ресурс] – Режим доступа : http://www.intuit.ru/
2. Электронная библиотечная система «IPRbooks» [Электронный ресурс] – Режим доступа : http://www.iprbookshop.ru/

### 8.7. Методические указания по освоению дисциплины

*Методические указания при работе над конспектом во время проведения лекции*

В ходе лекционных занятий необходимо вести конспектирование учебного материала. Общие и утвердившиеся в практике, правила и приемы конспектирования лекций:

Конспектирование лекций ведется в специально отведенной для этого тетради, каждый лист которой должен иметь поля, на которых делаются пометки из рекомендованной литературы, дополняющие материал прослушанной лекции, а также подчеркивающие особую важность тех или иных теоретических положений.

Необходимо записывать тему и план лекций, рекомендуемую литературу к теме. Записи разделов лекции должны иметь заголовки, подзаголовки, красные строки. Для выделения разделов, выводов, определений, основных идей можно использовать цветные карандаши и фломастеры.

Названные в лекции ссылки на первоисточники надо пометить на полях, чтобы при самостоятельной работе найти и вписать их.

В конспекте дословно записываются определения понятий, категорий и законов. Остальное должно быть записано своими словами.

Каждому обучающемуся необходимо выработать и использовать допустимые сокращения наиболее распространенных терминов и понятий.

В конспект следует заносить всё, что преподаватель пишет на доске, а также рекомендуемые схемы, таблицы, диаграммы и т.д.

*Методические указания по подготовке к практическим и лабораторным работам*

Целью практических и лабораторных работ является углубление и закрепление теоретических знаний, полученных обучающимися на лекциях и в процессе самостоятельного изучения учебного материала, а, следовательно, формирование у них определенных умений и навыков.

В ходе подготовки к практическим и лабораторным работамнеобходимо прочитать конспект лекции, изучить основную литературу, ознакомиться с дополнительной литературой, выполнить выданные преподавателем задания. При этом учесть рекомендации преподавателя и требования программы. Дорабатывать свой конспект лекции, делая в нем соответствующие записи из литературы. Желательно при подготовке к практическим и лабораторным работампо дисциплине одновременно использовать несколько источников, раскрывающих заданные вопросы.

*Методические указания по организации самостоятельной работы*

Самостоятельная работа приводит обучающегося к получению нового знания, упорядочению и углублению имеющихся знаний, формированию у него профессиональных навыков и умений.

Самостоятельная работа выполняет ряд функций:

- развивающую;
- информационно-обучающую;
- ориентирующую и стимулирующую;
- воспитывающую;
- исследовательскую.

Виды самостоятельной работы, выполняемые в рамках курса:

1. Проработка и повторение лекционного материала
2. Подготовка к практическим занятиям
3. Реферат
4. Подготовка к аттестации

Обучающимся рекомендуется с самого начала освоения курса работать с литературой и предлагаемыми заданиями в форме подготовки к очередному аудиторному занятию. При этом актуализируются имеющиеся знания, а также создается база для усвоения нового материала, возникают вопросы, ответы на которые обучающийся получает в аудитории.

Можно отметить, что некоторые задания для самостоятельной работы по курсу имеют определенную специфику. При освоении курса обучающийся может пользоваться библиотекой вуза, которая в полной мере обеспечена соответствующей литературой. Значительную помощь в подготовке к очередному занятию может оказать имеющийся в учебно-методическом комплексе краткий конспект лекций. Он же может использоваться и для закрепления полученного в аудитории материала.

*Методические указания по работе с литературой*

Всю литературу можно разделить на учебники и учебные пособия, оригинальные научные монографические источники, научные публикации в периодической печати. Из них можно выделить литературу основную (рекомендуемую), дополнительную и литературу для углубленного изучения дисциплины.

Изучение дисциплины следует начинать с учебника, поскольку учебник – это книга, в которой изложены основы научных знаний по определенному предмету в соответствии с целями и задачами обучения, установленными программой.

При работе с литературой следует учитывать, что имеются различные виды чтения, и каждый из них используется на определенных этапах освоения материала.

Предварительное чтение направлено на выявление в тексте незнакомых терминов и поиск их значения в справочной литературе. В частности, при чтении указанной литературы необходимо подробнейшим образом анализировать понятия.

Сквозное чтение предполагает прочтение материала от начала до конца. Сквозное чтение литературы из приведенного списка дает возможность обучающемуся сформировать свод основных понятий из изучаемой области и свободно владеть ими.

Выборочное – наоборот, имеет целью поиск и отбор материала. В рамках данного курса выборочное чтение, как способ освоения содержания курса, должно использоваться при подготовке к лабораторным практикумам по соответствующим разделам.

Аналитическое чтение – это критический разбор текста с последующим его конспектированием. Освоение указанных понятий будет наиболее эффективным в том случае, если при чтении текстов обучающийся будет задавать к этим текстам вопросы. Часть из этих вопросов сформулирована в приведенном в ФОС перечне вопросов для

собеседования. Перечень этих вопросов ограничен, поэтому важно не только содержание вопросов, но сам принцип освоения литературы с помощью вопросов к текстам.

Целью изучающего чтения является глубокое и всестороннее понимание учебной информации.

Есть несколько приемов изучающего чтения:

1. Чтение по алгоритму предполагает разбиение информации на блоки: название; автор; источник; основная идея текста; фактический материал; анализ текста путем сопоставления имеющихся точек зрения по рассматриваемым вопросам; новизна.

2. Прием постановки вопросов к тексту имеет следующий алгоритм:

- медленно прочитать текст, стараясь понять смысл изложенного;
- выделить ключевые слова в тексте;
- постараться понять основные идеи, подтекст и общий замысел автора.

3. Прием тезирования заключается в формулировании тезисов в виде положений, утверждений, выводов.

К этому можно добавить и иные приемы: прием реферирования, прием комментирования.

Важной составляющей любого солидного научного издания является список литературы, на которую ссылается автор. При возникновении интереса к какой-то обсуждаемой в тексте проблеме всегда есть возможность обратиться к списку относящейся к ней литературы. В этом случае вся проблема как бы разбивается на составляющие части, каждая из которых может изучаться отдельно от других. При этом важно не терять из вида общий контекст и не погружаться чрезмерно в детали, потому что таким образом можно не увидеть главного.

## 9. МАТЕРИАЛЬНО-ТЕХНИЧЕСКОЕ ОБЕСПЕЧЕНИЕ ДИСЦИПЛИНЫ

Для реализации дисциплины требуется следующее материально-техническое обеспечение (специальные помещения):

- для проведения занятий лекционного типа

учебная аудитория, оснащенная учебной мебелью, оборудованная проектором, ПК, экраном, доской.

- для проведения занятий семинарского типа, практических занятий

учебная аудитория, оснащенная учебной мебелью, оборудованная проектором, ПК, экраном, доской.

- для проведения текущего контроля и промежуточной аттестации

учебная аудитория, оснащенная учебной мебелью, оборудованная проектором, ПК, экраном, доской.

- для групповых и индивидуальных консультаций

учебная аудитория, оснащенная учебной мебелью, оборудованная проектором, ПК, экраном, доской.

- для самостоятельной работы:

помещение, оснащенное компьютерной техникой с возможностью подключения к сети Интернет и обеспечением доступа в электронную информационно-образовательную среду Института

## 10. ОСОБЕННОСТИ ОСВОЕНИЯ ДИСЦИПЛИНЫ ЛИЦАМИ С ОГРАНИЧЕННЫМИ ВОЗМОЖНОСТЯМИ ЗДОРОВЬЯ

Обучающимся с ограниченными возможностями здоровья предоставляются специальные учебники, учебные пособия и дидактические материалы, специальные технические средства обучения коллективного и индивидуального пользования, услуги ассистента (тьютора), оказывающего обучающимся необходимую техническую помощь, а также услуги сурдопереводчиков и тифлосурдопереводчиков.

Освоение дисциплины обучающимися с ограниченными возможностями здоровья может быть организовано совместно с другими обучающимися, а также в отдельных группах.

Освоение дисциплины обучающимися с ограниченными возможностями здоровья осуществляется с учетом особенностей психофизического развития, индивидуальных возможностей и состояния здоровья.

В целях доступности получения высшего образования по образовательной программе лицами с ограниченными возможностями здоровья при освоении дисциплины обеспечивается:

1) для лиц с ограниченными возможностями здоровья по зрению:

– присутствие тьютора, оказывающий студенту необходимую техническую помощь с учетом индивидуальных особенностей (помогает занять рабочее место, передвигаться, прочитать и оформить задание, в том числе, записывая под диктовку),

– письменные задания, а также инструкции о порядке их выполнения оформляются увеличенным шрифтом,

– специальные учебники, учебные пособия и дидактические материалы (имеющие крупный шрифт или аудиофайлы),

– индивидуальное равномерное освещение не менее 300 люкс,

– при необходимости студенту для выполнения задания предоставляется увеличивающее устройство;

2) для лиц с ограниченными возможностями здоровья по слуху:

– присутствие ассистента, оказывающий студенту необходимую техническую помощь с учетом индивидуальных особенностей (помогает занять рабочее место, передвигаться, прочитать и оформить задание, в том числе, записывая под диктовку),

– обеспечивается наличие звукоусиливающей аппаратуры коллективного пользования, при необходимости обучающемуся предоставляется звукоусиливающая аппаратура индивидуального пользования;

– обеспечивается надлежащими звуковыми средствами воспроизведения информации;

3) для лиц с ограниченными возможностями здоровья, имеющих нарушения опорно-двигательного аппарата:

– письменные задания выполняются на компьютере со специализированным программным обеспечением или надиктовываются тьютору;

– по желанию студента задания могут выполняться в устной форме.

**Приложение 1**

# ФОНД ОЦЕНОЧНЫХ СРЕДСТВ (ОЦЕНОЧНЫЕ МАТЕРИАЛЫ) ДЛЯ ПРОВЕДЕНИЯ ТЕКУЩЕГО КОНТРОЛЯ УСПЕВАЕМОСТИ И ПРОМЕЖУТОЧНОЙ АТТЕСТАЦИИ

**по дисциплине «Разработка мобильных приложений»**

## 1. Показатели и критерии оценки результатов освоения дисциплины

| Результаты обучения (код и наименование) | Показатель оценивания | Критерии оценивания | Процедуры оценивания |
|---|---|---|---|
| ПК-2 Способность разрабатывать и адаптировать прикладное программное обеспечение | | | |
| Знает современные объектно-ориентированные языки программирования | Демонстрация знаний современных объектно-ориентированных языков программирования | Полнота и системностьзнаний о современных объектно-ориентированных языках программирования | устный опрос |
| Умееткодировать на языках программирования; тестировать результаты прототипирования; верифицировать структуру программного кода | Демонстрация умений кодирования на языках программирования; тестирования результатов прототипирования; верифицирования структуры программного кода | полнота и правильность выполнения практического задания | Практические задания |
| Владеетнавыками разработки структуры программного кода; верификации структуры программного кода относительно архитектуры ИС и требований заказчика к ИС; разработки руководства программиста ИС; обеспечения соответствия разработанного кода и процесса кодирования на языках программирования принятым в организации или проекте стандартам и технологиям. | Демонстрация навыков владения разработки структуры программного кода; верификации структуры программного кода относительно архитектуры ИС и требований заказчика к ИС; разработки руководства программиста ИС; обеспечения соответствия разработанного кода и процесса кодирования на языках программирования принятым в организации или проекте стандартам и технологиям. | полнота и правильность выполнения практического задания | Практические задания |

| | | | |
|---|---|---|---|
| ПК-8 Способность проводить тестирование компонентов программного обеспечения ИС; | | | |
| Знаетинструменты и методы модульного тестирования; регламенты модульного тестирования. | освоение теоретических основ тестирования компонентов программного обеспечения ИС | полнота и правильностьтр актовки теоретических основ тестирования компонентов программного обеспечения ИС | устный опрос, практические задания |
| Владеетнавыками обеспечения соответствия процессов модульного тестирования ИС принятым в организации или проекте стандартам и технологиям. | Применение информационно-коммуникационных технологий с учетом основных требований тестирования компонентов программного обеспечения ИС | полнота и правильность выполнения практического задания | Практические задания |
| ПК-2<br>ПК-8 | | | Промежуточная аттестация: экзамен |

## 2. Методические материалы, определяющие процедуры оценивания

### 2.1. Методические материалы, определяющие процедуры оценивания в рамках текущего контроля успеваемости

**Устные опросы** проводятся во время лекций, практических занятий и возможны при проведении промежуточной аттестации в качестве дополнительного испытания при недостаточности результатов тестирования. Основные вопросы для устного опроса доводятся до сведения студентов на предыдущем занятии.

Количество вопросов определяется преподавателем.

Время проведения опроса от 10 минут до 1 академического часа.

Устные опросы строятся так, чтобы вовлечь в тему обсуждения максимальное количество обучающихся в группе, проводить параллели с уже пройденным учебным материалом данной дисциплины и смежными курсами, находить удачные примеры из современной действительности, что увеличивает эффективность усвоения материала на ассоциациях.

Критерии и шкала оценки устного опроса

Развернутый ответ студента должен представлять собой связное, логически последовательное сообщение на заданную тему, показывать его умение применять определения, правила в конкретных случаях.

**«отлично»** ставится, если:

1) студент полно излагает материал, дает правильное определение основных понятий;

2) обнаруживает понимание материала, может обосновать свои суждения, применить знания на практике, привести необходимые примеры не только из учебника, но и самостоятельно составленные;

3) излагает материал последовательно и правильно с точки зрения норм литературного языка.

**«хорошо» -** студент дает ответ, удовлетворяющий тем же требованиям, что и для «отлично», но допускает 1–2 ошибки, которые сам же исправляет, и 1–2 недочета в последовательности и языковом оформлении излагаемого.

**«удовлетворительно»** – студент обнаруживает знание и понимание основных положений данной темы, но:

1) излагает материал неполно и допускает неточности в определении понятий или формулировке правил;

2) не умеет достаточно глубоко и доказательно обосновать свои суждения и привести свои примеры;

3) излагает материал непоследовательно и допускает ошибки в языковом оформлении излагаемого.

**«неудовлетворительно»** ставится, если студент обнаруживает незнание большей части соответствующего вопроса, допускает ошибки в формулировке определений и правил, искажающие их смысл, беспорядочно и неуверенно излагает материал. Оценка «2» отмечает такие недостатки в подготовке, которые являются серьезным препятствием к успешному овладению последующим материалом.

**Практические задания** выполняются студентами на практических занятиях. Студентам необходимо выполнить практические задания, указанные преподавателем. Результаты работы сохранить в файлах. После выполнения заданий необходимо преподавателю продемонстрировать результаты работы и быть готовым ответить на вопросы и продемонстрировать выполнение отдельных пунктов заданий. Защита выполненных практических заданий осуществляется на практическом занятии.

Критерии и шкала оценки практических заданий

**«отлично»** ставится, если: студент самостоятельно и правильно решил учебно-профессиональную задачу, уверенно, логично, последовательно и аргументировано излагал свое решение, используя изученные понятия.

**«хорошо»** ставится, если: студент самостоятельно и в основном правильно решил учебно-профессиональную задачу, уверенно, логично, последовательно и аргументировано излагал свое решение, используя изученные понятия.

**«удовлетворительно»** ставится, если: студент в основном решил учебно-профессиональную задачу, допустил несущественные ошибки, слабо аргументировал свое решение, используя в основном изученные понятия.

**«неудовлетворительно»** ставится, если: студент не решил учебно-профессиональную задачу.

**Методические материалы, определяющие процедуры оценивания в рамках промежуточной аттестации**

Промежуточная аттестация по дисциплине проводится в форме устного **экзамена** по расписанию экзаменационной сессии.

Вопросы к экзамену доводятся до сведения студентов заранее.

Билет к экзамену содержит 2 вопроса.

При подготовке к ответу пользование учебниками, учебно-методическими пособиями, средствами связи и электронными ресурсами на любых носителях запрещено.

Время на подготовку ответа – от 30 до 45 минут.

По истечении времени подготовки ответа, студент отвечает на вопросы экзаменационного билета. На ответ студента по каждому вопросу билета отводится, как правило, 3-5 минут.

После ответа студента преподаватель может задать дополнительные (уточняющие) вопросы в пределах предметной области экзаменационного задания.

После окончания ответа преподаватель объявляет обучающемуся оценку по результатам экзамена, а также вносит эту оценку в экзаменационную ведомость, зачетную книжку.

Критерии и шкала оценки экзамена

**«отлично»** ставится, если:

- студент глубоко и всесторонне усвоил программный материал;
- уверенно, логично, последовательно и грамотно его излагает;
- опираясь на знания основной и дополнительной литературы, тесно привязывает усвоенные научные положения с практической деятельностью;
- умело обосновывает и аргументирует выдвигаемые им идеи;
- делает выводы и обобщения;
- свободно владеет системой понятий по дисциплине.

**«хорошо»** ставится, если:

- студент твердо усвоил программный материал, грамотно и по существу излагает его, опираясь на знания основной литературы;
- не допускает существенных неточностей;
- увязывает усвоенные знания с практической деятельностью бакалавра;
- аргументирует научные положения;
- делает выводы и обобщения;
- владеет системой понятий по дисциплине.

**«удовлетворительно»** ставится, если:

- студент усвоил только основной программный материал, по существу излагает его, опираясь на знания только основной литературы;
- допускает несущественные ошибки и неточности;
- испытывает затруднения в практическом применении знаний;
- слабо аргументирует научные положения;
- затрудняется в формулировании выводов и обобщений;
- частично владеет системой понятий по дисциплине.

**«неудовлетворительно»** ставится, если:

- студент не усвоил значительной части программного материала;
- допускает существенные ошибки и неточности при рассмотрении проблем;
- испытывает трудности в практическом применении знаний;
- не может аргументировать научные положения;
- не формулирует выводов и обобщений.

## 3. Типовые контрольные задания

### Типовые задания для текущего контроля успеваемости

#### 3.1 Типовые вопросы для устного опроса при текущем контроле

4. Каковы аппаратные требования к устройствам WindowsPhone?
5. В чём преимущество использования сенсорного экрана емкостного типа по сравнению с резистивным?
6. Какие устройства WindowsPhone позволяют телефону определять своё местоположение?
7. Какие аппартные кнопки есть у устройств WindowsPhone и какие функции

они выполняют?

8. Какие типы сетевых подключений поддерживаются в WindowsPhone?

9. С какими программами и службами может взаимодействовать WindowsPhone? В чём идея "быстрого переключения приложений"?

10. Каковы отличия фоновых задач от программ?

11. Как выполняется компиляция и запуск программы в WindowsPhone? Для чего используется эмулятор WindowsPhone?

12. Как приложение может использовать функции телефона? Для чего используются технологии Silverlight и XNA?

13. Какие средства для хранения данных есть в WindowsPhone?

14. Какие инструменты можно использовать для создания приложений для WindowsPhone?

15. Длячегонужен Windows Phone Marketplace?

16. Какие средства разработки можно использовать для создания интерфейса приложений?

17. Что такое Metro-стиль?

18. Для чего предназначены свойства и методы?

19. Как создать интерфейс страницы приложения Silverlight?

20. Как можно изменить внешний вид элементов Silverlight на странице приложения? Для чего используется язык XAML?

21. Как обрабатывать события в приложении Silverlight?

22. Как можно создать на основе файлов исходного кода программы исполняемый

23. Как в VisualStudio создать новую программу? Что такое "пространство имен"?

24. Как совместно использовать несколько проектов? Каким образом можно добавить в проект ресурсы? Что такое "динамическая библиотека"?

25. Что такое "построение программы"?

26. Чем отличается термин "решение" от термина "проект"?

27. Какие типы решений и проектов приложений для WindowsPhone поставляются cWindowsPhone SDK?

28. Что представляют собой файлы с расширением .xap?

29. Как в VisualStudio запустить отладку приложения в эмуляторе WindowsPhone? Какие средства VisualStudio позволяют осуществлять отладку приложений?

30. Как в программе можно обработать ошибку, если пользователь введет вместо числа произвольный текст?

31. Как во время работы программы изменить цвет текста?

32. Что нужно сделать, чтобы при выборе пользователем текстового поля выводилась специальная клавиатура для ввода чисел?

33. Как при создании программы можно изменить свойства элемента Silverlight?

34. Как вывести на экран сообщение, чтобы приложение могло получить ответ от пользователя?

35. Какие настройки можно задать при добавлении в проект ресурса? В чем разница между элементом контента и внедренным ресурсом?

36. Что нужно сделать, чтобы добавить к программу код, который должен выполняться, когда происходит определенное событие?

37. Для чего используется привязка данных?

38. Как выполнить привязку объекта данных к визуальному элементу Silverlight?

39. В чем разница между однонаправленной и двунаправленной привязкой

данных? Как можно указать поддерживаемый страницей приложения тип ориентации?

40. Для чего предназначены объекты-контейнеры? Как можно вывести в приложение список данных?

41. Как осуществить переход к другой странице приложения? Как можно передать данные другой странице приложения? Как создать и использовать в приложении класс ViewModel? Для чего используются наблюдаемые коллекции?

42. Где приложение может сохранять данные?

43. Каким образом программа может сохранять файлы?

44. Как программа может сохранить настройки, которые можно представить в виде пар "имя—значение"?

45. Для чего предназначена программа IsolatedStorageExplorer?

46. Как можно создать базу данных в приложении для WindowsPhone?

47. Как связать результаты запроса LINQ с визуальными элементами Silverlight?

48. Как с помощью LINQ задать связи между таблицами?

49. Как выполняются запросы LINQ?

50. Как выполняется добавление и удаление записей таблиц в LINQ?

51. Какие типы подключений к сети доступны в WindowsPhone?

52. Какие образом данные передаются по сети?

53. Для чего используется система доменных имён?

54. Каким образом несколько взаимодействующих с сетью служб могут работать на одном сервере только с теми сообщениями, которые для них предназначены?

55. В чём отличие сеансов подключений от дейтаграмм?

56. Для чего используется класс Socket?

57. Как в программе можно создать подключение по протоколу UDP?

58. Как в программе можно создать подключение по протоколу TCP?

59. Для чего используется класс WebClient?

60. Как получить необходимые данные из XML-структуры с помощью LINQ?

61. Для чего предназначена технология WindowsCommunicationFoundation? Как создать службу WCF?

62. Как создать приложение для WindowsPhone для использования методов службы WCF?

63. Для чего предназначена технология XNA? В чём отличие решений XNA от Silverlight?

64. Какие действия выполняются при загрузке игр XNA? Как в программе используется контент-менеджер?

65. Как в программе создать и вывести на экран графический объект?

66. Как можно использовать сенсорный экран для управления игровыми объектами? Какие действия необходимо выполнить для вывода текста на экран приложения? Каково назначение и принцип работы акселерометра?

67. Как использовать акселерометр в приложениях для WindowsPhone? Какие классы XNA используются для воспроизведения звуков в игре? Как управлять воспроизведением звука в игре?

68. Как воспроизводить звук в программе Silverlight?

69. Как в приложении можно управлять настройками экрана?

70. Каким образом можно совместно использовать технологии Silverlight и XNA в одном приложении?

71. Какие изображения должно содержать приложение для WindowsPhone, и как эти изображения используются?

72. Как можно создать и использовать в приложении экран-заставку?

73. Что означает "быстрое переключение приложений"?

74. Как происходит переключение приложений?

75. Как можно сохранить состояние приложения при его деактивации и загрузить при возобновлении работы?

76. Как в приложении вызвать системные функции телефона?

77. Чем отличаются задачи запуска и задачи выбора?

78. Как приложение может использовать фоновые задачи?

79. В чём разница между периодическими и ресурсоёмкими задачами? Как создать и использовать в приложении агента фоновой задачи?

80. Какие возможности есть у инструмента для анализа производительности приложений для WindowsPhone?

81. Для чего нужны файлы манифеста?

82. Какие значки и изображения необходимо подготовить перед распространением приложения через WindowsPhoneMarketplace?

83. Для чего нужна разблокировка телефона?

84. Как проходит процесс одобрения приложения?

85. Для чего предназначен тестовый режим приложения?

86. Как организовать закрытое бета-тестирование приложения?

## 3.3. Типовые практические задания

### Общие сведения о платформе WindowsPhone 7.5

Ваша компания планирует создать приложение для адвокатов. Приложение должно хранить и управлять информацией о времени, которое тратит их персонал на работу с клиентами. Менеджер вашей компании побеседовал с потенциальными покупателями продукта и предоставил следующую информацию.

Система будет использоваться юридическим штатом, чтобы поминутно отслеживать действия сотрудников.

Первоочередные задачи системы — предоставление актуальной информации, её надёжное хранение и простота использования.

Программа будет подключаться к нашим серверам тайм-менеджмента и загружать расписания работы и информацию о клиентах в телефон в начале каждого дня.

В течение рабочего дня пользователи будут вводить информацию о своих действиях, и в конце дня телефон должен загружать эту информацию обратно на сервер.

Периодически будут возникать неотложные случаи, которым необходимо уделять внимание, и наша система должна отправлять сообщение сотрудникам о возникновении подобных ситуаций.

Иногда персонал будет участвовать во встречах за пределами офиса, и при этом система также должна отслеживать их действия.

Ваша компания планирует использовать в телефонах ещё несколько приложений, включая юридический словарь и интерактивную юридическую систему. Новое приложение должно работать одновременно с этими программами.

Мы также хотели бы использовать телефонные устройства в качестве диктофона, чтобы делать аудио- и видеозаписи встреч с клиентами с высоким качеством. К записям будут добавляться теги, и они будут загружаться на наши сервера. Записи будут занимать примерно 100 Мб данных за один час.

Наше приложение должно работать совместно с другими приложениями в устройстве, которые клиенты могут загрузить.

Компания иногда должна отправлять информационное "сообщение дня" с корпоративного веб-сайта. Приложение должно принимать и выводить эти сообщения на экран. В идеале, это должно происходить, даже когда приложение не запущено.

В будущем может понадобиться отслеживать местоположение тех сотрудников, которые выезжают за пределы офиса.

Ваш менеджер попросил вас исследовать платформу WindowsPhone как потенциальное устройство для выполнения поставленных задач. Вы должны предоставить ответы на следующие вопросы:

Могут ли возникнуть какие-либо проблемы при использовании телефона для решения этих задач?

На основе какой технологии следует создать приложение: Silverlight или XNA?

Какой тип сетевого подключения должен использоваться для распределения расписания и получения отчётов?

Следует ли ограничить владельцев телефонов возможностью запуска только одного нашего приложения?

Достаточно ли в телефоне памяти для хранения данных для обслуживания 10 встреч в день?

Есть ли в WindowsPhone какие-либо особенности, которые можно использовать для дальнейшего улучшения приложения?

Есть ли у платформы какие-либо ограничения, которые в будущем могут привести к изменениям технических характеристик устройств?

Какой язык программирования вы предлагаете использовать для разработки приложения? Можно ли использовать некоторые библиотеки для обработки данных, которые некоторое время назад были написаны на VisualBasic .NET?

Есть ли какие-то особые требования для создания и выполнения фоновых задач в телефоне?

Можно ли передать какие-то операции в облачный сервер?

При ответе на каждый вопрос вы должны определить подходящие средства и возможности платформы WindowsPhone.

**Введение в Silverlight**

1. Пользовательский интерфейс программы Калькулятор времени Ваша компания решила создать приложение для адвокатов. Приложение должно

хранить и управлять информацией о времени, которое тратит их персонал на работу с клиентами. В настоящее время адвокаты записывают в отчет количество минут, которые они потратили на работу с клиентом, и эти отчеты обрабатываются вручную. После того как адвокатам выдали телефоны на платформе WindowsPhone, было решено создать простой Калькулятор времени, который они смогут использовать для расчета времени.

Адвокаты будут вводить время запуска и время окончания (в часах и минутах), и программа должна выводит на экран количество минут между этими отметками времени.

Ваш менеджер попросил Вас разработать пользовательский интерфейс Silverlight для этого приложения:

Определите, какие элементы интерфейса Silverlight нужно разместить на главной странице приложения.

Нарисуйте схему расположения элементов на форме. Создайте "Руководство пользователя" для приложения.

Попытайтесь определить все возможные случаи ввода недопустимых значений. Добавьте описание действий приложения в случае ввода недопустимых данных.

Создайте набор тестовых данных и результатов, которые должна выдать программа, и оформите их в виде таблицы. Количество тестов должно быть не менее 10.

2. Пользовательский интерфейс калькулятора времени

Один из менеджеров по продукции считает, что телефон должен отправлять детализированную информацию о каждом временном интервале в корпоративную

систему так, чтобы информация о проведенном времени могла обновляться автоматически. Менеджер решил, что нужно отправлять следующую информацию:

дата и время начала работы; дата и время окончания работы; название компании-клиента;

местоположение клиента, если встреча происходит не в помещении компании клиента;

оценка качества работы, поставленная клиентом — целое число в диапазоне от одного до пяти.

На сервер нужно отправлять отчеты, содержащие указанную информацию о каждой рабочей встрече.

Создайте пример XML-файла, содержащий описания трех встреч.

Удостоверьтесь, что файл содержит записи о встречах в помещении компании-заказчика, а также записи о встречах за пределами компании.

**Управление решениями в VisualStudio**

1. Создание приложения Калькулятор времени

Создание пустого приложения Silverlight

Запустите VisualStudio с установленным WindowsPhone SDK.

В главном меню выберите пункт Файл -> Создать проект…. Откроется диалоговое окно Создать проект.

В списке установленных шаблонов выберите SilverlightforWindowsPhone. В центральной части окна выберите Приложение WindowsPhone.

В нижней части окна в поле Имя введите TimeCalculator.

Выберите подходящий каталог для проекта. По умолчанию VisualStudio поместит проект в папку VisualStudio 2010 \ Projects в папке Документы.

Убедитесь, что флажок Создать каталог для решения установлен, и нажмите OK, чтобы создать новое приложение.

VisualStudio попросит выбрать целевую платформу для нового приложения. Выберите ОС WindowsPhone 7.1 и нажмите OK. VisualStudio создаст новое приложение в указанном местоположении.

Нажмите клавишу F5, чтобы запустить пустую программу. Запустится эмулятор WindowsPhone (если он еще не был запущен), и приложение будет выполняться в эмуляторе.

Щелкните по окну VisualStudio, чтобы его выбрать. Нажмите клавиши Shift + F5, чтобы остановить выполнение программы. Также можно остановить программу, нажав кнопку Назад в эмуляторе WindowsPhone.

Добавление элементов Silverlight

Убедитесь, что в VisualStudio открыто окно MainPage.xaml. Слева находится окно дизайнера (оно похоже на WindowsPhone), в центре — код XAML, справа — обозреватель решений. (Указанные области могут располагаться в окне VisualStudio по- другому.)

Измените текст имя страницы в элементе PageTitle на TimeCalculator. Измените текст МОЕ ПРИЛОЖЕНИЕ в элементе ApplicationTitle на ваше имя.

В главном меню выберите пункт Вид -> Панель элементов, чтобы отобразить на экране панель инструментов с доступными элементами. Элементы можно перетаскивать с панели инструментов в окно дизайнера.

Добавьте следующие элементы пользовательского интерфейса в окно дизайнера приложения и задайте их имена, как указано в таблице:

| Назначениеэлемента | Типэлемента | Имяэлемента |
|---|---|---|
| | | |

| |
|---|
| Часы времени начала TextBoxstartHourTextBox<br>Минуты времени начала TextBoxstartMinTextBox<br>Часы времени окончания TextBoxendHourTextBox Минуты времени окончания TextBoxendMinTextBox Кнопка для вычисления Кнопка calcButton<br>РезультатTextBlockresultTextBlock |

Можно также добавить дополнительные элементы TextBlock, чтобы разместить в окне приложения пояснительный текст.

Настройте свойства добавленных элементов, чтобы приложение выглядело приблизительно таким образом (рекомендуется установить размер текста элементов TextBlock равным 25):


Запустите программу. Обратите внимание, что можно вводить текст в текстовые поля и нажимать на кнопку.

Добавление поведения в приложение

В обозревателе решений откройте файл MainPage.xaml.cs.

Добавьте после конструктора MainPage следующее описание метода calculateMinutes:

privatevoidcalculateMinutes()

{

// здесь нужно будет добавить проверку данных и обработку исключений

int startMin = int.Parse(startMinTextBox.Text); int startHour = int.Parse(startHourTextBox.Text);

int endMin = int.Parse(endMinTextBox.Text); int endHour = int.Parse(endHourTextBox.Text);

// здесь нужно будет добавить код для вычисления времени

int result = 99;

resultTextBlock.Text = "Результат: " + result.ToString();

}

Вернитесь в окно редактора файла MainPage.xaml.

Дважды щелкните мышью по кнопке вычислить. VisualStudio создаст обработчик нажатия на кнопку и откроет код обработчика, созданный в файле MainPage.xaml.cs.

private void calcButton_Click(object sender, RoutedEventArgs e)

{

}

Добавьте вызов метода calculateMinutes в этот метод-обработчик.

Запустите программу. Обратите внимание, что при нажатии на кнопку значение результата изменяется на 99.

Завершение создания приложения

В редакторе кода XAML добавьте во все текстовые поля следующий атрибут: InputScope="Number"

Это укажет WindowsPhone, что в текстовые поля будут вводиться числа.

Добавьте в метод calculateMinutes код для вычисления разности времени между введенными моментами времени (создайте код самостоятельно).

Запустите программу и проверьте правильность ее работы. Упражнение 2. Отладка приложения

В этом упражнении вы выполните отладку приложения Калькулятор времени, которое создал другой программист. Программа иногда выдает неправильные результаты. Необходимо выполнить отладку программы, чтобы обнаружить проблему.

Откройте проект TimeCalculator, который находится в папке Lab3 TimeCalculator. Выполните программу. Введите значения Начало: 00:00 и Окончание: 01:00.

Нажмите кнопку вычислить. Обратите внимание, что программа выводит на экран 60 минут, что является правильным результатом.

Введите значения Начало: 00:00 и Окончание: 01:30.

Нажмите кнопку вычислить. Обратите внимание, что программа выводит на экран 60 минут, что является неверным результатом.

Во время работы программы откройте исходный файл MainPage.xaml.cs в обозревателе решений.

Установите точку останова в следующей строке, щелкнув в поле слева от этой строки:

resultTextBlock.Text = "Результат: " + result.ToString();

Нажмите кнопку вычислить еще раз. Теперь программа должна приостановить работу в этой строке.

Просмотрите значения переменных. Попробуйте установить причину возникающей проблемы. Если вы не можете обнаружить ошибку, выполните следующий шаг.

Нажмите клавишу F5 для возобновления программы. Введите значения Начало: 01:02 и Окончание: 03:04. Теперь используйте отладчик, чтобы просмотреть значения переменных startHour, startMin, endHour и endMin.

Определите и исправьте причину ошибок.

Удалите контрольную точку, щелкнув мышью в левом поле еще раз, и проверьте правильность работы программы на тестовых данных.

**Создание приложений Silverlight**

1. Добавление обработки ошибок

Существующая версия программа Калькулятор времени работает при вводе правильных числовых значений, но выдает исключение, если пользователь введет

значения, лежащие вне диапазона, или произвольный текст, который не является числом. Необходимо добавить в программу код для обработки следующих ошибок:

ввод произвольного текста вместо чисел;

ввод значения часа, которое больше 23 или меньше 0; ввод значения минут, которое больше 59 или меньше 0; ввод времени начала, которое больше времени окончания.

В этом упражнении Вы улучшите версию программы, созданную другим программистом.

Откройтев Visual Studio проектTimeCalculatorвпапке Lab4 TimeCalculator.

Запустите программу. Введите значения Начало: 0a:00 и Окончание: 00:00 и нажмите кнопку вычислить.

Приложение сгенерирует исключение, поскольку значение 0a не может быть преобразовано в число.

Измените код программы, чтобы учесть приведенные выше замечания.

Используйте метод TryParse для преобразования текста в число, чтобы программа могла обнаружить недопустимые значения.

Выделите на экран недопустимые значения красным цветом, а допустимые значения — цветом текста по умолчанию.

Выведите на экран окно с сообщением, если время начала и окончания события недопустимо.

Упражнение 2. Улучшение пользовательского интерфейса

Элементы TextBox в приложении могут генерировать событие TextChanged, когда пользователь вводит новые значения. Связанный с этими событиями код может автоматически обновлять значения на экране.

Добавьте обработчик события TextChanged так, чтобы не нужно было использовать кнопку вычислить.

Удалите кнопку вычислить.

Лучший способ это сделать заключается в том, чтобы создать один метод, который будет вызываться для отображения нового значения результата при изменении любого текстового поля. Этот метод можно использовать в обработчиках элементов TextBox.

Упражнение 3. Использование привязки данных

Другой программист создал класс TimeClass для приложения. Вклассепятьсвойств:

StartHourStartMinuteEndHourEndMinuteMinuteDifference

Вам нужно выполнить привязку данных для связывания элементов на экране с этими свойствами. Первые четыре свойства будут использовать двунаправленную привязку, а пятое свойство является выходным значением, которое используется для вывода на экран результата.

Добавление класса TimeClass в качестве ресурса

Сначала нужно добавить TimeClass в проект так, чтобы его могли использовать элементы Silverlight.

Откройтев Visual Studio проектTimeCalculatorв Lab4 Data Binding TimeCalculator. Откройте страницу MainPage.xaml.

Сначала нужно добавить пространство имен к ресурсам страницы MainPage. После строки, которая начинается на

xmlns:mc = "http://schemas.openxmlformats....добавьте строку

xmlns:local = "clr-namespace:TimeCalculator"

Теперь нужно добавить связь к классом, который мы хотим использовать. Послестрокиshell:SystemTray.IsVisible="True">добавьтестроки

<phone:PhoneApplicationPage.Resources>

<local:TimeClass x:Key="TimeClass" />

</phone:PhoneApplicationPage.Resources>

Теперь можно добавить ресурс к элементу Grid, который содержит элементы

Silverlight пользовательского интерфейса приложения. В элемент Grid с именем LayoutRoot добавьте следующий атрибут:

DataContext = "{StaticResourceTimeClass}" Связывание данных со свойствами элементов

В редакторе VisualStudio щелкните по элементу startHourTextBox. В области свойств будут отображаться свойства этого элемента.

Щелкните правой кнопкой мыши по свойству Text, и выберите в контекстном меню пункт Применить привязку данных…. Появится список доступных свойств.

Дважды щелкните по элементу StartHour. Обратите внимание, что в параметрах задан режим TwoWay.

Повторите эти действия для привязки свойств StartMin, EndHour и Endmin к соответствующим элементам.

В настройках привязки элемента resultTextBlock к свойству MinuteDifference установите режим привязки OneWay.

Запустите программу и введите значение времени окончания. Проверьте, что значение результата обновляется, когда пользователь переходит к другому элементу.

Теперь программа использует привязку данных, но часть обработка ошибок при вводе не выполняется. Попробуйте добавить в программу код для обработки ошибок. Для этого можно добавить в класс TimeClass свойства с сообщениями об ошибке и связать их с визуальными элементами.

Упражнение 4. Связывание данных со списками

Вашему начальнику нужно приложение для работы с информацией о клиентах, которое может выводить на экран список встреч с каждым клиентом. Программист начал писать эту программу, но не закончил ее. Вам поручено дописать программу.

Класс Session

Этот класс содержит текстовое описание встречи и ее продолжительность в минутах.

```
public class Session
{
public string Description { get; set; } public int LengthInMins { get; set; }

public Session(string inDescription, int inLength)
{
Description = inDescription; LengthInMins = inLength;
}
}
```

Класс Customer содержитсписоквстречсклиентом. public List<Session> Sessions;

Программа создает тестовый набор встреч для каждого клиента, но Ваш начальник считает, что этого недостаточно.

Увеличение количества тестовых данных

Для начала нужно изменить метод MakeTestCustomers класса Customers, чтобы создать больше тестовых данных.

Откройтев Visual Studio проектCustomerManagerвпапке Lab4 CustomerManager. ОткройтефайлCustomers.cs.

НайдитевклассеCustomers метод MakeTestCustomers. Этот метод создает тестовый набор клиентов, и для каждого клиента создается несколько тестовых встреч. Количество встреч для каждого клиента определяется случайным образом, но диапазон значений является слишком маленьким.

Найдитестроку

```
intnoOfSessions = sessionRand.Next(1,4);
```

Метод Next возвращает случайное значение из диапазона 1—3, что позволит создать максимум 3 тестовых встречи с одним клиентом.

Измените параметры метода так, чтобы для каждого клиента было создано 20—30 встреч. Это позволит проверить, что выводимый на экран список клиентов можно прокручивать.

Добавление навигации по страницам

Страница с информацией о клиенте теперь содержит кнопку встречи, которая используется для просмотра встреч с клиентом. Но пока кнопке не назначен обработчик событий, и при ее нажатии ничего не происходит. Необходимо создать обработчик нажатия на кнопку и добавить код для перехода к странице встреч.

Откройте в редакторе VisualStudio страницу CustomerDetailPage.xaml.

Дважды щелкните по кнопке встречи, чтобы создать обработчик событий и перейти к файлу кода CustomerDetailPage.xaml.cs.

В созданный метод sessionButton_Click добавьте следующий код для выполнения перехода на страницу.

NavigationService.Navigate(new Uri("/SessionDetailPage.xaml", UriKind.RelativeOrAbsolute));

Запустите программу. Выберите клиента и нажмите на кнопку встречи. Произойдет переход на страницу встреч, но на экран ничего не будет выводиться, потому что к странице не привязаны никакие данные.

Добавление привязки данных на страницу Sessions

Откройте в VisualStudio файл с исходным кодом SessionDetailPage.xaml.cs.

Переместите курсор в класс сразу после закрывающей фигурной скобки конструктора SessionDetailsPage. Введите слово override и нажмите пробел.

Intellisense выведет на экран список методов, которые могут быть переопределены в этом классе. Выберите метод OnNavigatedTo и нажмите Enter.

Добавьте следующий код в метод OnNavigatedTo.

// получить ссылку на страницу, содержащую информацию о выбранном клиенте AppthisApp = Application.CurrentasApp;

CustomerName.DataContext = thisApp.ActiveCustomer; SessionList.ItemsSource = thisApp.ActiveCustomer.Sessions;

Запустите программу. Выберите клиента и нажмите на кнопку встречи. На экран будет выведен список встреч для выбранного клиента. При нажатии на кнопку Назад произойдет переход на страницу информации о клиенте. Если нажать кнопку Назад еще раз, откроется страница со списком клиентов.

**Хранение данных приложений**

1. Использование изолированного хранилища

В предыдущем упражнении мы создавали простое приложение TimeTracker, которое сохраняло информацию о времени встреч с клиентами. Предыдущая версия программы при запуске генерировала набор тестовых данных. Необходимо использовать хранилище файлов, чтобы приложение сохраняло информацию о встречах.

Загрузка сохраненных данных

Программист внес в программу изменения, однако, тестировщик обнаружил, что программа не сохраняет изменения данных. Необходимо выявить причину этой проблемы.

Откройте в VisualStudio проект CustomerManager в папке Lab5 CustomerTimeLoggerStorage. Этот проект содержит версию программы, в которой обнаружена ошибка.

Убедитесь в том, что эмулятор WindowsPhone не запущен. Укажите эмулятор WindowsPhone в качестве целевой платформы.

Запустите программу. При первом запуске программа создаст новый набор тестовых данных.

Выберите клиента и измените информацию о нем.

Нажмите кнопку сохранить. Обратите внимание на то, что содержимое экрана изменилось, поскольку в программе используется привязка данных для отображения обновленных значений.

Остановите программу, нажав в эмуляторе WindowsPhone кнопку Назад. Не останавливайте выполнение программы в VisualStudio.

Запустите программу еще раз. При повторном запуске программа должна загрузить список клиентов из изолированного хранилища.

Найдите клиента, информацию о котором вы изменили. Обратите внимание, что сделанные изменения не сохранились.

Список клиентов загружается в файле App.xaml.cs. Откройте этот файл в обозревателе решений VisualStudio и найдите конструктор App. В конце этого метода находится код, который должен загружать данные из изолированного хранилища. Попробуйте определить причину возникающей проблемы.

Конструктор содержит только код, который создает тестовый список при каждом запуске программы:

ActiveCustomerList = Customers.MakeTestCustomers();

Необходимо изменить этот код, чтобы информация о клиентах загружалась из файла. Откройте в файле App.xaml.cs область CustomerManagerValues и найдите методы LoadCustomers и SaveCustomers.

Метод LoadCustomers возвращает список клиентов или значение null, если список не может быть считан. Если метод возвращает null, программа должна создать набор тестовых данных. Замените указанный в шаге 10 код на следующий:

// загрузить список клиентов из файла ActiveCustomerList = LoadCustomers(ListFilename);

// если загрузить список не удалось, создать тестовые данные if (ActiveCustomerList == null)

ActiveCustomerList = Customers.MakeTestCustomers(); Запуститепрограмму. Измените информацию о любом клиенте.

Остановите программу, нажав в эмуляторе WindowsPhone кнопку Назад. Не останавливайте выполнение программы в VisualStudio.

Запустите программу еще раз. Убедитесь в том, что теперь сделанные изменения были сохранены.

Пропадание информации о встрече

Тестировщик нашел в программе еще одну проблему: программа некорректно сохраняет информацию о встрече. Необходимо исследовать эту проблему.

Программа создает объекты для работы с потоками StreamReader и StreamWriter для загрузки и сохранения элементов в изолированном хранилище. Каждый объект может сохранить свое состояние в поток и загрузить информацию из потока. Ниже приведен код методов для сохранения информации в поток и для загрузки информации из потока:

```
public void SaveToStream(StreamWriter output)
{
output.WriteLine(CustomerList.Count);

foreach (Customer c in CustomerList)
```

```
{
c.SaveToStream(output);
}
}

public Customers(StreamReader input)
{
CustomerList = new List<Customer>();
int noOfCustomers = int.Parse(input.ReadLine()); for (int i = 0; i<noOfCustomers; i++)
{
CustomerList.Add(new Customer(input));
}
}
```

Этот код выглядит правильно. Необходимо проверить класс Customer, правильно ли он использует эти методы.

Вернитесь в программу и откройте файл Customers.cs. Найдите класс Customers и просмотрите метод SaveToStream: publicvoidSaveToStream(StreamWriteroutput)

```
{
output.WriteLine(Name); output.WriteLine(Address); output.WriteLine(ID);
}
```

Этот метод не содержит код для сохранения информации о встрече. Аналогичная проблема возникает и в методе Customer:

```
public Customer(StreamReader input)
{
Name = input.ReadLine(); Address = input.ReadLine();
ID = int.Parse(input.ReadLine());
int noOfSessions = int.Parse(input.ReadLine());
}
```

Необходимо использовать тот же шаблон для загрузки и сохранения списка встреч, который использует объект CustomerList для хранения информации о клиентах. Измените методы для сохранения и загрузки, чтобы можно было сохранять информацию о нескольких встречах:

```
public void SaveToStream(StreamWriter output)
{
output.WriteLine(Name); output.WriteLine(Address); output.WriteLine(ID); output.WriteLine(Sessions.Count); foreach (Session session in Sessions)
{
session.SaveToStream(output);
}
}

public Customer(StreamReader input)
{
Name = input.ReadLine(); Address = input.ReadLine();
ID = int.Parse(input.ReadLine());
int noOfSessions = int.Parse(input.ReadLine()); for (int i = 0; i<noOfSessions; i++)
{
Sessions.Add(new Session(input));
}
}
```

Остановите эмулятор WindowsPhone, чтобы при следующем запуске программы она создала новый набор тестовых данных. Запустите программу еще раз.

Остановите программу, нажав в эмуляторе WindowsPhone кнопку Назад. Не останавливайте выполнение программы в VisualStudio.

Запустите программу еще раз.

Выберите любого клиента и просмотрите информацию о встречах, которая должна была быть сохранена.

Упражнение 2. Использование базы данных

В программу, созданную в предыдущем упражнении, внесли изменения, чтобы она могла работать с базой данных. Однако, программа компилируется, но не запускается. Необходимо исправить ошибку.

Откройте в VisualStudio проект CustomerManager в папке Lab5 CustomerTimeLoggerDatabase. Этот проект содержит версию программы, в которой обнаружена ошибка.

Убедитесь в том, что эмулятор WindowsPhone не запущен.

Запустите программу. Программа сгенерирует тестовые данные и попытается их использовать. После этого будет сгенерировано исключение, в котором база данных сообщает, что проблема связана с внешним ключом. Это касается связей между классами Session и Customer. Сообщение об ошибке информирует о том, что невозможно вставить внешний ключ, так как не существует соответствующий первичный ключ.

При создании связи встречи с клиентом запись в таблице встреч должна содержать значение первичного ключа, которое идентифицирует клиента. Однако, если этот ключ еще не был задан, база данных не может добавить запись. При создании связи между этими двумя таблицами необходимо:

добавить встречу к списку встреч клиента;

установить значение customerID для встречи, чтобы идентифицировать клиента для этой встречи.

Код, который выполняет эти действия, выглядит следующим образом: intsessionLength = sessionRand.Next(5,120);

string sessionDesc = "Встреча " + i.ToString(); Session newSession = new Session(); newSession.Description = sessionDesc; newSession.LengthInMins = sessionLength; newSession.SessionCustomer = c; newDB.SessionTable.InsertOnSubmit(newSession); c.Sessions.Add(newSession);

Добавьте этот код в программу и запустите ее повторно. Теперь программа работает правильно и позволяет управлять данными.

Тестирование хранения данных

В настоящее время при каждом запуске программа создает новую базу данных. Измените программу так, чтобы она использовала существующую базу данных и создавала новую, если база данных не существует. Обратите внимание, что новая база данных создается при выполнении следующей строки кода в файле App.xaml.cs:

CustomerDB.MakeTestDB("Data Source=isostore:/Sample.sdf");

**Средства WindowsPhone для работы с сетью**

Для выполнения необходимо работающее сетевое подключение, а также среда VisualStudio для создания службы WCF.

1. Создание службы TimeTracker

В этом упражнении вы создадите службу, которая получает и хранит отчёты о встречах, получаемые от приложения TimeTrackerClient, работающего на удалённом устройстве. Служба будет предоставлять два метода: один — для получения отчётов о встречах, другой — для предоставления отчётов о встречах.

Создание службы Откройте VisualStudio.

Выберите в главном меню пункт Файл -> Создать проект. Откроется диалоговое окно Создать проект.

В списке Установленные шаблоны выберите в группе WCF шаблон Приложение службы WCF.

Назовите проект TimeTrackerService и нажмите OK, чтобы создать проект. VisualStudio создаст новый проект и откроет файл Service1.svc.cs.

Задайте службе и её интерфейсу более удобные имена. Для этого выберите в обозревателе решений файл IService1.cs, нажмите F2 и введите новое имя ITimeTrackerService.cs. При нажатии клавиши EnterVisualStudio предложит переименовать интерфейс и все ссылки на него, чтобы его имя соответствовало новому имени файла. Нажмите OK для подтверждения.

Аналогичным образом переименуйте файл службы от Service1.svc в TimeTrackerService.svc.

Откройте файл TimeTrackerService.svc.cs и щёлкните правой кнопкой по названию класса Service1. В контекстном меню выберите пункт Рефакторинг -> Переименовать…, укажите в открывшемся окне новое имя класса TimeTrackerService и подтвердите переименование.

Создание интерфейсов методов службы

Наша служба будет содержать два метода: один метод будет использовать удалённый клиент для сохранения отчёта о встрече, а другой будет возвращать список сохранённых встреч. VisualStudio автоматически создаёт некоторые методы, которые мы удалим из проекта.

Откройте файл ITimeTrackerService.cs. Он содержит интерфейс, который описывает методы, предоставляемые службой.

Удалите весь код внутри интерфейса и добавьте следующее описание методов: [OperationContract]

void SaveSession(string employeeName, string companyName, int startHour, int startMin,

int endHour, int endMin);

код:

[OperationContract] string GetSessionList();

УдалитеописаниеклассаCompositeType. Создание методов службы

Откройте файл TimeTrackerService.svc.cs.

Удалите всё содержимое класса TimeTrackerService и добавьте в него следующий

class Session

{

public string EmployeeName{ get; set; } public string CompanyName { get; set; } public int StartHour { get; set; }

public int StartMin{ get; set; } public int EndHour { get; set; } public int EndMin { get; set; }

public override string ToString()

{

return EmployeeName + " " + CompanyName;

}

}

static List<Session> sessions = new List<Session>();

Этот класс будет хранить информацию о каждой встрече, а также содержит список встреч, который будет формироваться во время работы службы. В реальном приложении этот список должен заполняться информацией из базы данных.

Добавьте в класс TimeTrackerService следующие методы:

```
public void SaveSession(string employeeName, string companyName, int startHour, int startMin, int endHour, int endMin)
{
sessions.Add(new Session{
EmployeeName = employeeName, CompanyName = companyName, StartHour = startHour,
StartMin = startMin, EndHour = endHour, EndMin = endMin
});
}

public string GetSessionList()
{
StringBuilder result = new StringBuilder();

for (int i = 0; i<sessions.Count; i++)
{
result.AppendLine(sessions[i].ToString());
}
returnresult.ToString();
}
```

Метод SaveSession сохраняет информацию о сеансе на сервере. Метод GetSessionList возвращает сохранённую информацию о встречах в виде многострочного текста.

Развёртывание службы

Выберите в обозревателе решений файл TimeTrackerService.svc и нажмите F5 для запуска проекта. После построения решения откроется окно тестового клиента WCF.

Дважды щёлкните мышью по методу SaveSession(). Введите следующие значения параметров метода:

employeeName — Test Employee companyName — Test Company startHour — 1

startMin — 2

endHour —3

endMin — 4

Нажмите на кнопку Вызвать, чтобы вызвать метод службы с указанными параметрами. Метод вернёт пустое значение.

Дважды щёлкните мышью по методу GetSessionList() и нажмите на кнопку Вызвать. Метод вернёт строку, соответствующую встрече, добавленной в систему при вызове метода SaveSession.

Нажмите клавиши Shift + F5 для остановки службы.

В обозревателе решений щёлкните правой кнопкой мыши по файлу TimeTrackerService.svc и выберите в контекстом меню пункт Просмотр в обозревателе. Обратите внимание на URL службы в адресной строке браузера — это значение понадобится для подключения к службе клиента:

http://localhost:30975/TimeTrackerService.svc

В вашем случае номер порта может отличаться, поскольку VisualStudio генерирует его автоматически.

Закройте браузер и запустите службу в VisualStudio ещё раз. Запущенная служба будет использоваться в следующем упражнении.

Упражнение 2. Создание клиента службы

В этом упражнении мы создадим приложение, которое будет подключаться к созданной в предыдущем упражнении службе и использовать её методы.

Подключение клиентского приложения к службе

Убедитесь в том, что созданная в предыдущем упражнении служба запущена.

Запустите другую копию VisualStudio и откройте в ней проект TimeTrackerClient в папке Lab6 TimeTrackerClient. В текстовые поля пользователь может ввести имя сотрудника, название компании, с которой проводится встреча, а также время начала и окончания встречи. При нажатии на кнопку сохранить, программа должна сохранить информацию о встрече на сервере, и при нажатии на кнопку список встреч — вывести информацио о встречах, полученную от службы.

В обозревателе решений щёлкните правой кнопкой мыши по элементу Ссылки и выберите пункт Добавить ссылку на службу….

В открывшемся диалоговом окне Добавить ссылку на службу в поле Адрес введите URL службы, полученный в предыдущем упражнении в результате запуска службы, и нажмите кнопку Перейти. VisualStudio загрузит описание службы с сервера и выведет на экран информацию о службе.

Откройте описание службы, щёлкнув по стрелке слева от её имени, и выберите имя интерфейса ITimeTrackerService. В поле операции будут отображены имена двух созданных методов службы.

Измените пространство имён на TimeTrackerService и нажмите кнопку OK. VisualStudio добавит в проект ссылку на службу.

Щёлкните правой кнопкой мыши по созданной ссылке на службу TimeTrackerService и выберите пункт Настроить ссылку на службу….

В открывшемся диалоговом окне снимите галочку Повторно использовать типы в сборках, на которые есть ссылки и нажмите OK.

Использование метода службы SaveSession в приложении Откройте файл MainPage.xaml.cs.

Добавьте в начало объявления класса следующую строку: privateTimeTrackerService.TimeTrackerServiceClienttimeTracker;

Добавьте в конструктор класса после вызова метода InitializeComponent следующий код для создания связанного со службой объекта и обработчиков событий службы (для создания обработчиков событий можно использовать Intellisense — в этом случае автоматически будет создана большая часть кода):

```
timeTracker = new TimeTrackerService.TimeTrackerServiceClient();

timeTracker.SaveSessionCompleted +=
new EventHandler<System.ComponentModel.AsyncCompletedEventArgs>(
timeTracker_SaveSessionCompleted);

timeTracker.GetSessionListCompleted +=
new EventHandler<TimeTrackerService.GetSessionListCompletedEventArgs>(
timeTracker_GetSessionListCompleted);
```

Добавьте код метода saveButton_Click для асинхронного вызова метода службы и передачи введённых пользователем значений:

```
private void saveButton_Click(object sender, RoutedEventArgs e)
{
int startHour, startMin, endHour, endMin;
```

if (int.TryParse(startHourTextBox.Text, out startHour) &&int.TryParse(startMinTextBox.Text, out startMin) &&int.TryParse(endHourTextBox.Text, out endHour) &&int.TryParse(endMinTextBox.Text, out endMin))

{

timeTracker.SaveSessionAsync(employeeNameTextBox.Text,

companyNameTextBox.Text, startHour, startMin, endHour, endMin);

}

}

Добавьте код метода-обработчика timeTracker_SaveSessionCompleted, который будет вызываться при завершении выполнения метода службы:

void timeTracker_SaveSessionCompleted(object sender, System.ComponentModel.AsyncCompletedEventArgs e)

{

if (!e.Cancelled)

{

employeeNameTextBox.Text = "Введитеимясотрудника..."; companyNameTextBox.Text = "Введитеназваниекомпании...";

}

}

Если метод службы выполнится успешно, то текст в текстовых полях приложения будет изменён на первоначальный.

Тестирование метода SaveSession

Нажмите F5, чтобы запустить программу в эмуляторе.

Введите в текстовые поля информацию о встрече и нажмите на кнопку сохранить.

Текст в текстовых полях должен измениться на первоначальный. Таким же образом добавьте информацию о второй встрече. Остановите выполнение программы.

Использование метода службы GetSessionList

Добавьте в приложение код метода listButton_click для вызова метода службы: privatevoidlistButton_Click(objectsender, RoutedEventArgs e)

{

timeTracker.GetSessionListAsync();

}

Добавьте в приложение код метода timeTracker_GetSessionListCompleted для вывода на экран результата выполнения метода службы:

void timeTracker_GetSessionListCompleted(object sender, TimeTrackerService.GetSessionListCompletedEventArgs e)

{

if (!e.Cancelled)

{

sessionsTextBlock.Text = e.Result;

}

}

Запустите программу и нажмите кнопку список встреч. На экран будет выведена информация о двух введённых встречах.

**Создание приложений XNA**

Для выполнения рекомендуется использовать физическое устройство WindowsPhone для возможности мультисенсорного ввода и использования акселерометра.

1. Создание многопользовательской игры

В этом упражнении вы доработаете созданную на лекции игру. Ограничение движения игрового объекта

На текущий момент левая платформа может перемещаться за пределы экрана.

Необходимо предотвратить такую возможность.

Откройте в VisualStudio проект PongGame в папке Lab7 PongGame.

Добавьте в метод Draw следующий код, который не позволит левой платформе уйти за пределы экрана:

```
if (lPaddleY< 0)
{
lPaddleY = 0;
}

if (lPaddleY + lPaddleRectangle.Height>GraphicsDevice.Viewport.Height)
{
lPaddleY = GraphicsDevice.Viewport.Height - lPaddleRectangle.Height;
}
```

Добавьте аналогичный код для правой платформы. Запустите игру и проверьте правильность её работы. Добавление возможности играть вдвоём

На текущий момент правая платформа управляется компьютером. Благодаря возможности мультисенсорного ввода, можно добавить в игру возможность управления правой платформы вторым игроком.

Измените код для управления движением платформы следующим кодом для обработки нескольких одновременных касаний экрана:

```
TouchCollection touches = TouchPanel.GetState();

foreach (TouchLocation touch in touches)
{
if (touch.Position.Y>GraphicsDevice.Viewport.Height / 2)
{
lPaddleY = lPaddleY + lPaddleSpeed;
}
else
{
lPaddleY = lPaddleY - lPaddleSpeed;
}
}
```

Запустите программу и убедитесь в том, что управление платформой работает правильно.

Теперь нужно изменить программный код для управления правой платформой, чтобы ей мог управлять второй игрок. Для этого экран можно поделить на четыре части. При обнаружении касания в каждой из областей определяется направление перемещения соответствующей платформы. Условия принадлежности касания соответствующей области показаны на рисунке:


Измените код программы, чтобы управление обеими платформами осуществлялось с сенсорного экрана. (Решите эту задачу самостоятельно.)

Удалите из программы код, который автоматически управляет правой платформой. Запустите игру и проверьте правильность её работы.

Упражнение 2. Использование акселерометра для управления в игре

В этом упражнении вы создадите приложение, которое использует акселерометр WindowsPhone для определения положения телефона в пространстве. Существующая программа выводит на экран зелёный шарик, расположенный над красной точкой, которая расположена в центре экрана. Шарик должен управляться с помощью акселерометра.

Откройте в VisualStudio проект Level в папке Lab7 Level. Откройте файл Game1.cs.

Добавьте следующий код в конструктор класса Game1 для указания поддерживаемой в приложении ориентации телефона, размеров экрана и задания полноэкранного режима.

graphics.SupportedOrientations = DisplayOrientation.LandscapeLeft; graphics.PreferredBackBufferWidth = 480;

graphics.PreferredBackBufferHeight = 800; graphics.IsFullScreen = true;

Добавьте в проект ссылку на библиотеку Microsoft.Devices.Sensors.

Добавьте в файл Game1.cs ссылку на пространство имён Microsoft.Devices.Sensors.

Добавьте в метод Initialise метод следующий код для создания и запуска объекта для работы с акселерометром:

```
Accelerometer acc = new Accelerometer();

acc.ReadingChanged +=
new EventHandler<AccelerometerReadingEventArgs>(acc_ReadingChanged);

acc.Start();
```

Добавьте сразу после метода Initialise следующий код для определения обработчика события акселерометра:

```
Vector3 accVector = Vector3.Zero;

void acc_ReadingChanged(object sender, AccelerometerReadingEventArgs e)
{
accVector.X = (float)e.X; accVector.Y = (float)e.Y; accVector.Z = (float)e.Z;
}
```

Для управления движением шарика можно использовать значения, получаемые от акселерометра и сохраняемые в переменную accVector. Переменные bubbleX и bubbleY содержат координаты шарика, когда он находится в центре экрана.

В методе Update удалите следующие строки кода: bubbleRectangle.X = (int)(bubbleX + 0.5f); bubbleRectangle.Y = (int)(bubbleY + 0.5f);

и добавьте код для вычисления новых координат шарика, в которые он будет перемещаться:

float accBubbleX = bubbleX - (accVector.X * 200); float accBubbleY = bubbleY + (accVector.Y * 200);

bubbleRectangle.X = (int)(accBubbleX + 0.5f); bubbleRectangle.Y = (int)(accBubbleY + 0.5f);

Запустите программу и проверьте правильность её работы.

**Использование системных функций в приложениях**

1. Редактирование значков приложения

Создание приложения

Откройте VisualStudio, создайте новое приложение Silverlight и назовите его Icons.

Выберите в качестве целевого устройства эмулятор WindowsPhone и запустите программу, нажав клавишу F5. VisualStudio развернёт программу в эмуляторе и запустит её.

Нажмите в эмуляторе кнопку Назад, чтобы остановить программу. В эмуляторе откроется меню Пуск, содержащее единственную плитку, связанную с программой InternetExplorer.

Щёлкните по стрелке вправо в верхнем правом углу экрана эмулятора. Откроется список приложений эмулятора. Список содержит приложение Icons, которое мы только что создали, InternetExplorer и приложение Настройки эмулятора.

Щёлкните по значку приложения Icons, чтобы запустить это. Приложение запустится в эмуляторе.

Нажмите кнопку Назад, чтобы остановить приложение.

Эмулятор WindowsPhone сохраняет все запускаемые в нём программы в списке приложений, пока его работа не будет остановлена. Можно открыть в VisualStudio другой проект и запустить его в эмуляторе — приложение добавится в список приложений эмулятора.

Прикрепление приложения к меню Пуск Откройте в эмуляторе список приложений.

Нажмите и не отпускайте левую кнопку мыши на приложении Icons. Появится меню, которое позволяет удалить приложение из эмулятора или прикрепить приложение к меню Пуск.

В открывшемся меню выберите пункт на рабочий стол. Приложение Icons будет прикреплено к меню Пуск рядом с плиткой приложения InternetExplorer.

Приложение также можно переместить в другое место меню Пуск или удалить из меню Пуск.

Редактирование значка приложения

В обозревателе решений щёлкните правой кнопкой мыши по файлу ApplicationIcon.png.

В открывшемся контекстном меню выберите пункт Открыть с помощью…, чтобы вывести на экран список программ, которые могут использоваться для работы с выбранным файлом.

Выберите в списке программу Paint и нажмите OK. Откроется окно программы Paint, в котором будет открыто изображение из файла.

Измените изображение и закройте программу Paint, сохранив сделанные изменения.

В главном меню VisualStudio выберите пункт Построение -> Перестроить. VisualStudio заново создаст файлы программы и будет использовать изменённую версию значка.

Запустите программу. VisualStudio удалит старую версию программы с эмулятора и развернёт новую версию с изменённым значком.

Аналогичным образом измените изображения в файлах Background.png и SplashScreenImage.jpg. После повторного построения приложения будут изменены значок в меню Пуск и экран-заставка программы.

2. Исследование выгрузки приложения

В этом упражнении вы рассмотрите, как приложения выгружаются из памяти. Создание приложения

Откройте VisualStudio, создайте новый проект приложения Silverlight и назовите его Tombstone.

Откройте файл App.xaml.cs и найдите методы, которые запускаются, когда происходят события запуска, закрытия, выгрузки и возобновления работы приложения. Добавьте в методы код для выдачи диагностических сообщений:

```
// Код для выполнения при активации приложения (переводится в основной режим)
// Этот код не будет выполняться при первом запуске приложения privatevoidApplication_Activated(objectsender, ActivatedEventArgs e)
{
System.Diagnostics.Debug.WriteLine("Активировано");
}

// Код для выполнения при деактивации приложения (отправляется в фоновый режим)
// Этот код не будет выполняться при закрытии приложения
private void Application_Deactivated(object sender, DeactivatedEventArgs e)
{
System.Diagnostics.Debug.WriteLine("Деактивировано");
}

// Код для выполнения при закрытии приложения (например, при нажатии пользователем
// кнопки "Назад")
// Этот код не будет выполняться при деактивации приложения privatevoidApplication_Closing(objectsender, ClosingEventArgs e)
{
System.Diagnostics.Debug.WriteLine("Закрывается");
}
```

Выберите в качестве целевого устройства эмулятор WindowsPhone и запустите приложение.

Выберите в главном меню VisualStudio пункт Вид -> Вывод. Откроется окно вывода диагностических сообщений. Последним сообщением будет Запущено.

В эмуляторе щёлкните кнопку Назад, чтобы остановить программу. В окне вывода диагностических сообщений появится сообщение Закрывается.

Деактивация приложения Запустите программу в эмуляторе.

Нажмите в эмуляторе кнопку Пуск для перехода к экрану меню Пуск. В окне вывода появится сообщение Деактивировано.

Запустите в эмуляторе программу InternetExplorer.

Нажмите в эмуляторе кнопку Назад, чтобы вернуться на экран меню Пуск.

Нажмите в эмуляторе кнопку Назад ещё раз, чтобы вернуться на страницу приложения. В окне вывода появится сообщение Активировано.

Закройте приложение. Возобновление работы приложения Откройте файл App.xaml.cs.

Найдите метод Application_Activated и измените его следующим образом, чтобы по выводимому сообщению можно было определить, из какого состояния приложение возобновляет свою работу:

```
private void Application_Activated(object sender, ActivatedEventArgs e)
{
if (e.IsApplicationInstancePreserved)
{
System.Diagnostics.Debug.WriteLine("АктивированоизсостоянияБездействует");
}
else
```

{
System.Diagnostics.Debug.WriteLine("АктивированоизсостоянияВыгружено");
}
}

Выполните все действия предыдущей части упражнения. Обратите внимание, что приложение было активировано из состояния "бездействует" не выгружалось из памяти.

Выгрузка приложения из памяти

В VisualStudio можно настроить, чтобы приложение при деактивации автоматически выгружалось из памяти.

Убедитесь в том, что приложение не запущено.

В обозревателе решений щёлкните правой кнопкой мыши по файлу проекта и в открывшемся меню выберите пункт Свойства. Откроется окно свойств проекта.

Выберите в окне свойств вкладку Отладка и установите галочку Отметить как удаленный в результате деактивации во время.

Выполните пункт 3 предыдущей части упражнения. Обратите внимание, что приложение было активировано из состояния "выгружено".

Наиболее вероятно, что работа приложения будет возобновлена из состояния покоя. Однако, необходимо проверять поведение программы, если она всё же будет выгружена из памяти.

**Публикация приложений в WindowsPhoneMarketplace**

1. Использование инструмента для анализа производительности

Откройте в VisualStudio один из созданных ранее проектов приложения Silverlight или XNA.

В главном меню VisualStudio выберите пункт Отладка -> Начать анализ производительности WindowsPhone. Откроется окно параметров.

Выберите необходимые параметры или оставьте значения по умолчанию.

Выберите в качестве целевого устройства эмулятор WindowsPhone и нажмите ссылку Запустить приложение. Приложение запустится в эмуляторе, и VisualStudio начнёт собирать сведения о работе приложения.

Выполните в приложении какие-нибудь операции и закройте его, нажав в эмуляторе кнопку Назад, находясь в главном окне приложения. VisualStudio закончит сбор данных и откроет созданный файл с расширением .sap в виде диаграммы.

Проанализируйте диаграмму. Обратите внимание, в какие моменты были зафиксированы всплески активности приложения, и вспомните, какие действия вы выполняли в это время.

Упражнение 2. Использование инструмента MarketplaceTestKit

В этом упражнении вы выполните тестирование приложения с помощью инструмента MarketplaceTestKit.

Откройте в VisualStudio один из созданных ранее проектов приложения Silverlight или XNA.

Выберите в раскрывающемся списке правее от выбора целевой платформы пункт Release.

Выполните построение проекта, нажав кливишиCtrl + Shift + B.

В главном меню VisualStudio выберите пункт Проект -> Открыть в MarketplaceTestKit. Откроется окно инструмента MarketplaceTestKit.

На вкладке Автоматизированные тесты нажмите на кнопку Запуск тестов.

Просмотрите результаты теста. Попробуйте определить, почему приложение не прошло некоторые тесты, и попытайтесь устранить недостатки.

## Типовые задания для промежуточной аттестации

### 3.3.Типовые контрольные вопросы для устного опроса на экзамене

1. Мобильное программирование, платформы для разработки.
2. Система Windows Phone 7. Microsoft Visual Studio Express for Windows Phone.
3. Аппаратные средства устройств, поддерживающих WindowsPhone 7.
4. Проектирование программы Silverlight. Язык XAML.
5. Пример создания приложения Silverlight для WindowsPhone.
6. Управление решениями в VisualStudio. Проекты и решения в VisualStudio. Отладка программ.
7. Улучшение приложения. Изменение и отображение данных.
8. Управление ориентацией страницы приложения. Отображение списков данных.
9. Навигация по страницам приложения. Использование классов ViewModel.
10. Хранилище данных WindowsPhone. Базы данных в WindowsPhone.
11. Создание связей данных в LINQ.
12. Основные сведения о сетях
13. . Создание подключения по протоколу UDP.
14. Создание подключения по протоколу TCP.
15. Подключение к сетевому ресурсу.
16. Чтение данных из XML-потока с помощью LINQ.
17. Взаимодействие приложений с сетевыми службами.
18. Основные сведения о технологии XNA.
19. Создание игрового мира. Использование средств WindowsPhone в играх.
20. Совместное использование XNA и Silverlight.
21. Значки и экраны-заставки приложений для WindowsPhone. Быстрое переключение приложений.
22. Задачи запуска и задачи выбора. Фоновые задачи.
23. Подготовка приложения для продажи.
24. Распространение приложений и игр для WindowsPhone.



Подписано цифровой подписью: Шиянов Сергей Евгеньевич
Дата: 24.02.2021 16:24:04
Организация: Автономная некоммерческая организация высшего образования "Северо-Кавказский социальный институт"