Программа учебной дисциплины

"Научно-исследовательский семинар

«Разработка облачных и мобильных приложений на платформе Apple iOS»"

Research seminar «Cloud and Mobile Applications Development

for Apple iOS Platform»

Утверждена

Академическим советом ООП

Протокол № от «\_\_»\_\_\_\_\_20\_\_ г.

|  |  |
| --- | --- |
| Автор  | Александров Дмитрий Владимирович |
| Число кредитов  | 3 |
| Контактная работа (час.)  | 50 |
| Самостоятельная работа (час.)  | 64 |
| Курс  | 3 |
| Формат изучения дисциплины | без использования онлайн курса |

1. Course Description
2. Title of a Course

Cloud and Mobile Applications Development for Apple iOS Platform

The research seminar ‘Cloud and Mobile Applications Development for Apple iOS Platform’ is offered to students of Bachelor Program «Software Engineering» (area code 09.04.04), at the Faculty of Computer Science, the National Research University - Higher School of Economics/HSE. The course is classified as a discipline of choice (Design and Research Works unit / Research Seminar of 2018 – 2019 academic year’s working syllabus); it is a three module course, which is started in module #1. The duration of the course amounts to 50 practical training hours. In addition, 64 academic hours are set aside for self-studying of students. Mark check stipulated by the working syllabus consists of results of performing of practical and homework tasks and final examination given in the 3rd module.

The course covers the Swift 4.2 programming language and iOS application development basics. In particularly, main aspects include design and architecture, user interactions specifics, capabilities of Apple devices, etc. The main objective of the training course is also to examine and principles of iOS apps development, get familiar with platform, IDE, technologies, and frameworks for the development of iOS apps.

The students will learn how to analyze, design, and develop iOS apps, draft user interfaces and create the app prototypes.

This course addresses also the following questions:

* The course is aimed to help students to develop skills that will enable them to construct mobile apps of high quality, reliable, and that is reasonably easy to understand, modify and maintain.
* Is there a difference between apps, which earned millions, and the projects, which were not compensated even their development cost?
* How to integrate the mobile apps with a cloud backend?
* Is it possible to increase the possibility of success of a new mobile app?
* How to place a new mobile app in the App Store properly?
* How to integrate various development, ranking, feedback tools in mobile app construction?
1. Pre-requisites

It is presupposed that the students should be familiar with the basics of the object-oriented analysis and programming, the software development methodology. Java/C#/C++ programming experience is eligible.

- Programming,

- Design software,

- Research seminar "Component oriented programming".

1. Course Type

Optional.

1. Abstract

The research seminar “Cloud and Mobile Applications Development for Apple iOS Platform” is concerned the Swift programming language and IDE XCode for professional mobile apps development. It is based on the training materials (tutorials) from mobile apps developers for Apple iOS platform.

Modern mobile apps development requires the collaborative effort of a diverse team with varied skills. To be effective, team members need to understand the activities performed at each stage in the development cycle. In this course, students will analyze, design, and implement applications that meet user requirements through performing a collective team iOS app project. The course is aimed at students taking courses at mobile apps development in commerce and industry. It describes the technology and IDE, which are applied for iOS app development. The course is covering the main points of iOS apps development gives to the listeners to gather creation’s experience of real world apps on the basis of different prototypes.

2. Learning Objectives

During the three-module course “Cloud and Mobile Applications Development for Apple iOS Platform”, the students will:

* Study the Swift 4.2 programming language and XCode IDE for the fast development of iOS apps,
* Acquire skills in mobile apps design, development and implementation for Apple iOS platform,
* Focus on techniques for iOS app development and modern trends in that field.

The students will be able to answer the questions:

* Give a definition of terms «mobile app», «back-end», «cloud platform», etc. (the matter is related to main concepts introduced in the training course);
* What are the ideas of Agile way of development of mobile apps?
* Describe the IDE for iOS app development.
* What are the key challenges facing mobile app development? Provide examples.
* Describe mobile app design process and provide the examples.
* What are the principles of modern UI design for iOS apps?
* What are the trends in iOS app development?
* What are the main iOS app development process activities?
* What is the need of mobile app prototyping? What are the benefits and problems of prototyping?
* Describe the features of using the microservices approach for creating mobile apps.

3. Learning Outcomes

At the end of the course the students will acquire theoretical and practical knowledge and will be able to use:

- Technical skills

1. Develop and write a proposal for project of iOS app development.
2. Define an architecture of an iOS app.
3. Design a mobile app with back-end.
4. Implement iOS apps using Swift programming language and IDE XCode.
5. Prepare the final report for the course.
6. Discuss the processes of mobile apps development.

- Soft skills

1. Work within a team under mobile app project.
2. Be able to effectively communicate their work (Presentation skills).
3. Be able to write a project report (report writing skills).
4. Be able to reflect on their own work as well as the work of others (evaluate their experience).

4. Course plan

*Seminars*: 50 academic hours.

*Self-training*: 64 academic hours.

*Current control and the exam*: 11.5 academic hours.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| № | Topic Name | Course Hours, Total | Audience Hours | Self Study |
| Lectures | PracticalStudies |
| **Module 1. Apple’s Swift 4.2 Programming Language and iOS App Development** |
| 1 | Getting Started With Xcode and Swift Language. iOS Features: Controlling a Keyboard. | 6 |  | 2 | 4 |
| 2 | Work in the IDE Xсode. Visual design in the Storyboard. Using Autolayout. | 6 |  | 2 | 4 |
| 3 | Architectural Patterns in iOS (MVC, MVP, MVVM, VIPER). iOS Features: Timers, Tables, Permanent Storage and Web Content. | 6 |  | 2 | 4 |
| 4 | The Life Cycle of UIViewController. iOS Features: Animation, Audio, Gesture Recognition. | 7 |  | 3 | 4 |
| 5 | Maps & Geolocation. Core Data, Json Data & Webviews. Interaction with Integrated Software. | 8 |  | 3 | 5 |
|  | **Module 1, totally:** | **33** |  | **12** | **21** |
| **Module 2. iOS app development: design and implementation** |
| 6 | Development of Widgets and the Apps for Apple Watch with a General Container for Data Storage. Writing the Software Requirements Specifications (SRS).  | 8 |  | 4 | 4 |
| 7 | Extra Usefull Features. Advanced CoreLocation: using iBeacon. | 7 |  | 3 | 4 |
| 8 | Creating the iMessage Apps. AGILE Way of Mobile Apps Development.  | 7 |  | 3 | 4 |
| 9 | Creating the Apps that Use the Remote services. Wapor Framework.  | 7 |  | 3 | 4 |
| 10 | Using the Cloud Data Stores (IBM Bluemix and etc.) as a Backend for iOS Apps. | 8 |  | 3 | 5 |
|  | **Module 2, totally:** | **37** |  | **16** | **21** |
| **Module 3. iOS app development: design and implementation** |
| 11 | Creating the IoT Apps. | 9 |  | 5 | 4 |
| 12 | Creating the Chat Apps for Payments. | 9 |  | 5 | 4 |
| 13 | Creating the Games with Sprite Kit. | 8 |  | 4 | 4 |
| 14 | Extra Usefull Frameworks. ARKit, Core ML | 9 |  | 4 | 5 |
| 15 | Submitting an App To The App Store | 9 |  | 4 | 5 |
|  | **Module 3, totally:** | **44** |  | **22** | **22** |
|  | **TOTAL:** | **114** |  | **50** | **64** |

**Topic 1. Getting Started With Xcode and Swift**

♦ Topic outline:

* The Xcode Interface Overview
* Adding Labels To the Apps
* Working With Text Inputs and Buttons
* The Swift Playground Overview
* Variables And Variable Types
* Arrays And Dictionaries
* If and Else Statements
* For and While Loops
* Using Constraints for Creating Robust Layouts

**Topic 2. Work in the IDE Xсode**

♦ Topic outline:

* Navigation & Menu Bars With Bar Button Items
* Visual design in the Story-board.
* Creating Multiple View Controllers & Using Segues
* Executing Code Before & After Segues
* Using Autolayout.

**Topic 3. Architectural Patterns in iOS. iOS Features**

♦ Topic outline:

* Architectural Pattern MVC.
* Architectural Pattern MVP.
* Architectural Pattern MVVM.
* Architectural Pattern VIPER.
* Work with Timers
* Spinners And Alerts
* Adding The User Table
* Table Views & Working With Classes
* Storing Data Permanently In the Apps
* Viewing Website & Web Content In the Apps
* String Manipulation & Processing Web Data

**Topic 4. The Life Cycle of UIViewController. iOS Features**

♦ Topic outline:

* The Life Cycle of UIViewController.
* Working with Data From The Web
* Working with Images From The Web
* Getting Images From The Photo Library And Camera
* Manipulating UIImages and Creating Animations
* Working with Audio
* Responding To User Shaking and Swipes

**Topic 5. Maps & Geolocation**

♦ Topic outline:

* Integrating Maps Into the Apps and Adding Pins and Messages
* Geolocation: Using The User's Location In the Apps
* Working With Core Data and Searching For Objects
* Json Data
* Webviews
* Interaction with Integrated Software.
* Working With Json Data & Dictionaries

**Topic 6. Development of Widgets and the Apps for Apple Watch**

♦ Topic outline:

* Development of Widgets
* Development and the Apps for Apple Watch
* Using the General Container for Data Storage
* Writing the Software Requirements Specifications (SRS).

**Topic 7. Extra Usefull Features**

♦ Topic outline:

* Custom Keyboards
* Integrating iAds and In App Purchases Into the Apps
* Implementing Version Control
* 3D Touch: Peek & Pop and Quick Actions
* Implementing Touch ID
* Advanced CoreLocation: using iBeacon

**Topic 8. Creating the iMessage Apps**

♦ Topic outline:

* Creating the iMessage Apps
* AGILE Way of Mobile Apps Development

**Topic 9. Creating the Apps that Use the Remote services**

♦ Topic outline:

* Creating the Wapor Framework Project Integrated with the iOS Apps
* Creating the Apps that Use the Remote Services

**Topic 10. Using the Cloud Data Stores**

♦ Topic outline:

* Using the IBM Bluemix for Creation of a Backend for iOS Apps
* Using the other Cloud Data Stores for Creation of a Backend for iOS Apps

**Topic 11. Creating the IoT Apps**

♦ Topic outline:

* Creating the IoT App Prototypes for iOS and WatchOS

**Topic 12. Creating the Chat Apps for Payments**

♦ Topic outline:

* Using PassKit APIs to Support Apple Pay in iOS and WatchOS Apps
* Creating a Chat App Prototype for Payments

**Topic 13. Creating the Games with Sprite Kit**

♦ Topic outline:

* Adding And Animating Sprites
* Animating The Background
* Controlling and Spawning Sprites
* Collisions
* Scoring And Game Controls

**Topic 14. Extra Usefull Frameworks**

♦ Topic outline:

* Using the Framework ARKit to create unparalleled augmented reality experiences
* Using the Framework Core ML for Machine Learning

**Topic 15. Submitting an App To The App Store**

♦ Topic outline:

* App Icons And Launch Images
* Screenshots
* Submitting the Apps

5. Reading list

1. Required

1. The Swift Programming Language (Swift 4.2). – Apple Inc., 2014. – Режим доступа: <https://itunes.apple.com/us/book/the-swift-programming-language-swift-4-2/id881256329?mt=11>

2. [Dippery, Michael](https://ebookcentral.proquest.com/lib/hselibrary-ebooks/search.action?contributors=) Professional Swift. – John Wiley & Sons, Incorporated, 2015. – Режим доступа: <https://ebookcentral.proquest.com/lib/hselibrary-ebooks/reader.action?docID=4043081>

 3. Основы программирования на языке Objective-C для iOS : учеб. пособие / А.В. Кузин, Е.В. Чумакова. — М. : ИНФРА-М, 2019. — 118 с. — (Высшее образование: Бакалавриат). — Режим доступа: http://proxylibrary.hse.ru:2060/catalog/product/100326

1. Optional

1. Tyson McCann and Tyson McCann Art of the App Store : The Business of Apple Development. – Wrox, 2011. – Режим доступа: <https://ebookcentral.proquest.com/lib/hselibrary-ebooks/reader.action?docID=817945>

 2. Develop. – Apple Inc. – Режим доступа: <https://developer.apple.com/develop/>

 3. Human Interface Guidelines. – Apple Inc. – Режим доступа: <https://developer.apple.com/library/ios/documentation/userexperience/conceptual/mobilehig/LaunchImages.html#//apple_ref/doc/uid/TP40006556-CH22-SW1>!

6. Grading system

Students’ final grades are based on the following activities: reports on home assignment, in-class activities, and a final exam.

Ongoing assessment is delivered as *home assignment* (HA) and *in-class activities* (ICA). Grades, which are gained by students while ongoing assessment, are the parts of the *cumulative grade* (CG).

As far as home assignment consists of several parts, each part is finalized with a printed report. *Reports* on *home assignment* (RHAi, where i = 1,2,3) are used to evaluate the work over the duration of the course. Reports, which are received after a hard deadline, are evaluated as zero. If a product is not submitted to a store RHA2 is evaluated as Fail (1-3 at 10-point scale).

*In-class activities* (ICA) include evaluation of students attendance, involvement into discussions, quality of exercise performance during practical trainings and the end-of-module test at the end of 3rd module (see Section 7 for details). Only a student, who participates in-class activities gains ICA grade.

Repeating of in-class activities is not permitted, because ICA appears in CG.

Summarizing, ICA and RHAi form CG, which is calculated as follows:

CG = 0,5 \* ICA +0,2 \* RHA1 + 0,2 \* RHA2 + 0,1 \* RHA3

*Final exam* (FE) grade has two components: a theory grade (TG) (on-line test for Swift and iOS Application Development) and a practice grade (PG) (iOS mini project has to be implemented in Xcode on Swift).

FE = 0,5\*TG + 0,5\*PG

In general, final grade (FG) is calculated as follows:

FG = 0,2\*FE + 0,8\*CG

If a student verified the completion of one of MOOC listed below (Section 9), the extra credit (EC) is calculated. EC $\in $ [0;1] and calculates as a transformation of MOOCs grading scale to [0;1]. Consequently, the final grade includes extra credit and is calculated as follows:

FG = 0,2\*FE + 0,8\*CG + EC

Only rounded grades take place in calculations. The arithmetic rules are used. *Example, 4,5 -> 5; 4,49 -> 4.*

Attendance is graded according to 10-point scale applied in HSE.

|  |  |  |  |
| --- | --- | --- | --- |
| **10-point scale** | **Russian grading framework** | **ECTS grading scheme** | **5-point scale** |
| **10** | Excellent | A+ | Excellent | 5 |
| **9** | Excellent | A | Very good | 5 |
| **8** | Excellent | A– | Very good | 5 |
| **7** | Good | B+ | Good | 4 |
| **6** | Good | B– | Good | 4 |
| **5** | Satisfactory | C+ | Satisfactory | 3 |
| **4** | Satisfactory | C– | Satisfactory | 3 |
| **3** | Fail | F | Fail | 2 |
| **2** | Fail | F | Fail | 2 |
| **1** | Fail | F | Fail | 2 |

Final exam will have place at the end of 3rd module (the last module of the course).

**The question example of the on-line test for Swift**

Select the only one correct answer that will be received as a result of running the code (the number of points for this correct answer is 2):

class NamedShape {

    var numberOfSides: Int = 0

    var name: String

    init(name: String) {

        self.name = name

    }

}

class Square: NamedShape {

    var sideLength: Double

    init(sideLength: Double, name: String) {

        self.sideLength = sideLength

        super.init(name: name)

        numberOfSides = 4

    }

    func area() -> Double {

        return sideLength \* sideLength

    }

}

class EquilateralTriangle: NamedShape {

    var sideLength: Double = 0.0

    init(sideLength: Double, name: String) {

        self.sideLength = sideLength

        super.init(name: name)

        numberOfSides = 3

    }

    var perimeter: Double {

        get {

            return 3.0 \* sideLength

        }

        set {

            sideLength = newValue / 3.0

        }

    }

}

class TriangleAndSquare {

    var triangle: EquilateralTriangle {

        willSet {

            square.sideLength = newValue.sideLength

        }

    }

    var square: Square {

        willSet {

            triangle.sideLength = newValue.sideLength

        }

    }

    init(size: Double, name: String) {

        square = Square(sideLength: size, name: name)

        triangle = EquilateralTriangle(sideLength: size, name: name)

    }

}

var triangleAndSquare = TriangleAndSquare(size: 7, name: "another test shape")

triangleAndSquare.square = Square(sideLength: 23, name: "larger square")

print(triangleAndSquare.triangle.sideLength)

*What is the output? The possible answers are 23, 7 or 21.*

If the student has correctly answered 75 % of the questions, the test is considered as successfully passed.

**The practical part of the exam example**

Create the Mobile App that Uses the Yandex Translate API for the Translations of Small Texts.

**The home assignment or in-class assignment example**

 *Create a Sticker App*

We will build a message sticker extension that allows the users to send stickers while communicating with his/her friends in Messages. Apple has the App Store for iMessage, so we can sell our stickers and app extensions. To build an app extension for Messages, we will need to make use of the Message framework. Now we will focus on building a sticker pack. It doesn't need us to write any code.

*Prepare the Images.* Creating a sticker app is a two-part process: first, we prepare the images, that conforms to Apple's requirements; secondly, we create a sticker app project using Xcode.  Let's start with the first part. Messages supports various sticker image formats including PNG, GIF and JPG, with a maximum size of 500KB. It is recommended to use images in PNG format. Other than size, the other thing we have to consider, while preparing the images, is whether the images are static or animated. Messages supports both. Let's focus on the static ones. Now choose our own images and resize them to a size that best fits our stickers.

*Build the Sticker Package Project.* Assuming our images are ready, we build the sticker app. Create a new project in Xcode 8.3.3 or upper, choose iOS > Application and then select Sticker Pack Application. Next, fill in the project name. Click Next to continue and choose a folder to save the project. Xcode then generates the project.

*Add Images to the Sticker Pack.* Once the Xcode project is created, we'll see two folders in the project navigator. Click Stickers.xcstickers and then select the Sticker Pack folder. This is where we put our image files. Select all the images and drag them into the Sticker Pack folder. Now select the folder, and then choose the Attributes inspector. By default, the sticker size is set to Medium. The best choice is to set the sticker size to Small.

*Add App Icons.*Lastly, sticker pack must have an app icon. We prepare the icon with different sizes: 1024×768 points (@1x), 27×20 points (@1x, @2x, @3x), 32×24 points (@1x, @2x, @3x), 29×29 points (@1x, @2x, @3x), 60×45 points (@2x, @3x), 67×50 points (@1x, @2x), 74×55 points (@2x).

*Test the Sticker Pack.*Choose a simulated device and hit the Run button. Xcode will load the sticker pack into the Messages app and automatically launch it on the simulator. The Messages app in the simulator has come with two simulated users: Kate Bell and John Appleseed. The default user is set to Kate. To send a sticker to John, choose a sticker from the message browser and press return key to send it. Then go to John. We will find the stickers we've just sent.

7. Guidelines for Knowledge Assessment

In-class assignment during seminar hours is controlled by the use of keeping students attendance records and scoring (evaluation) involvement into discussions as well as quality of exercise performance during practical trainings.

The end-of-module test implies arrangement of the online written test (in lecture room) for all students enrolled to the course (the possibility of complementary interview with students is still reserved for the purpose of accurate grade confirmation). Subject area covered by the test embraces those topics of the course, which are discussed in seminars and practical trainings till the announced date; if student misses the end-of module test because of some valid reason, the grade «absence» is written down to corresponding column of the mark sheet. Depending on the number and complexity of the end-of-module test tasks (final decision is taken by the course instructor), the duration of the test amounts for 45 to 55 minutes. The end-of-module test is assessed on usual ten-point scale.

Home assignment is made as a capstone (individual) and team projects. Capstone and team projects will have the main result as a real world iOS apps with a backend on one of the popular cloud platforms.

There are some topics for presentations on the seminars:

* Agile Way of Mobile Apps Development;
* Innovative Models and Tools for Mobile App Development;
* Technologies and Frameworks for iOS Development;
* iOS App Development using Xamarin;
* iOS App Development using PhoneGap;
* Architecture of iOS Apps;
* Integrating iAds into iOS Apps;
* Integrating In App Purchases into iOS Apps;
* Using iBeacons in iOS Apps;
* Development of Geo Social iOS Apps;
* Submitting an App To The App Store;
* Patterns in iOS App Development;
* Trends in iOS App Development.

The minimum requirements to the capstone (individual) and the team projects:

- The capstone project must include a local database or use an external one.

- The team project must work with any backend with a database using the own or third-party framework or external resource (the cloud platform or the web service).

- One of the projects (capstone or team) must include the application for Apple Watch or TV.

- The Project should be provided by the documentation (Software Requirements Specification – SRS), operator's and programmer's guides).

For the all fully completed assignments (classroom and home), successfully passed test and implemented projects the maximum score can be 8 ("excellent"). If only one application will be published in the App Store, the maximum score can be 9. If the both (capstone and team projects) applications will be published in the App Store, the maximum score can be 10. If the student doesn't have any Apple Development Account, it will be provided to him or her. Maximum one point can be given for the victory in the iOS app development hackathon (or less than one point for just participation). One more extra point can be given for the registaration of the developed application (capstone or team) in Rospatent.

8. Methods of Instruction

The course combines direct and indirect teaching techniques and contains: presentations, collaborative learning, problem based learning, blended learning, reports, feedback and formative assessment.

*Direct teaching* is carried out by a seminar method. The preferred informal presentations (with discussion), where students play an active role. The main purpose of seminars is to introduce course’s topics, to overview basic information and to discuss and form the directions of further course activities.

*Collaborative* and *problem based learning* is implemented through *home assignments* and *in-class activity*.

*Home assignment* means that students work individually or in small collaborative teams ( 2 – 4 person) on a mobile app. Home project consists of three blocks: (1) conceiving the idea of a mobile app, domain and targeted audience studying; (2) developing an app and its submitting to the App Store (optionally but recommended); (3) the app monetizing and promotion (optionally but recommended). Moreover, home assignment is fully agreed with CDIO (Conceive-Design-Implement-Operate) initiative for engineering education.

*In-class activity* consists of short programming and project assignments. Students works individually or in pairs (*think share pare technique*) or in small randomly generated groups.

*Blended learning* is represented in two forms.

At first we use MOOCs to support students’ *self-training*. This course suggests students will complete MOOCs listed below to gain or improve their mobile apps developers’ skills.

At second we implement *assessment* using <http://letstest.ru/>.

The list of MOOCs, which may bring an EC and on-line resources:

|  |  |  |
| --- | --- | --- |
| **№ п/п** | **Name** | **Access conditions** |
| 1 |  **MOOC:** Developing iOS 11 Apps with Swift by Stanford University | [*https://itunes.apple.com/us/course/developing-ios-11-apps-with-swift/id1309275316*](https://itunes.apple.com/us/course/developing-ios-11-apps-with-swift/id1309275316) |
|  |  **Online resources:**ProQuest Ebook Central | *https://ebookcentral.proquest.com/lib/hselibrary-ebooks/home.action?ebraryDocId=null* |

9. Special Equipment and Software Support

Equipment for in-class and home activity:

1. iMac or MacBook with macOS Mojave10.14.1 or upper;
2. Classroom multimedia projector;
3. Apple iPhone and iPad (recommended);

Software recommended for in-class and home activity:

|  |  |  |
| --- | --- | --- |
| **№ п/п** | **Name** | **Access conditions** |
| 1 |  macOS Mojave10.14.1 or upper  | *Provided with iMacs* |
| 2 |  Xcode 10.0 or upper & Swift 4.2 or upper | *https://itunes.apple.com/ru/app/xcode/id497799835?mt=12* |