Введение

Языки программирования — интереснейшая область современной техники. За последние 30-40 лет информационные технологии разрослись до невероятных пределов, и сейчас мало кто в состоянии обозреть эту область в полном объёме. Компьютерные программы выросли с нескольких сотен строк до десятков миллионов строк, применяются сейчас в самых разных областях и запускаются на самых разных *платформах*, например:

- обыкновенные программы для персонального компьютера, часто называемые desktop-программами;
- web-программы, которые делятся, в свою очередь, на *клиентскую* часть, выполняющуюся на компьютере пользователя, и *серверную*, выполняющуюся на сервере;
- *мобильные* программы для планшетов, смартфонов и других мобильных устройств;
- *системные* программы, являющиеся частью операционной системы;
- *встраиваемые* программы, являющиеся частью встраиваемых систем управления (применяемые, например, в транспорте, банкоматах, станках с программным управлением, при программировании роботов).

Для написания разных видов программ сейчас применяются разные языки программирования. Например, в сфере *мобильных* программ сейчас правят бал языки Swift (мобильные устройства под управлением iOS), Java и Kotlin (устройства под управлением Android). *Системные* программы, как правило, пишутся на языках C или C++. Эти же языки долгое время использовались и для создания *встраиваемых* программ, но в последние годы в этой области набирает популярность язык Java. Для написания web-клиентов часто используется JavaScript, а в простых случаях — язык разметки страниц HTML. Web-серверы используют опять-таки Java (в сложных случаях), а также Python и PHP (в более простых). Наконец, простые desktop-программы сейчас могут быть написаны на самых разных языках, и выбор во многом зависит от сложности программы, области её использования, предполагаемой операционной системы. В первую очередь следует назвать языки Java, C++, C#, Python, Visual Basic, Ruby, Swift.

В каком-то смысле самым универсальным и одновременно самым распространённым языком программирования на данный момент следует считать язык Java. Java в широком смысле — не только язык, но и платформа для выполнения программ под самыми разными операционными системами и на разной аппаратуре. Такая универсальность обеспечивается наличием виртуальной машины Java — специальной системной программы, интерпретирующей Java байт-код в машинные коды конкретного компьютера или системы. Java также включает богатейший набор библиотек для разработки.

Однако для начинающих язык Java является несколько многословным и сложным. Это пособие посвящено другому языку программирования, спутнику Java — языку Котлин.

Котлин — молодой, лёгкий для изучения язык программирования, позволяющий писать программы под платформы JVM и Android более лаконично, просто и с меньшим количеством ошибок по сравнению с языком Java. Котлин и Java — полностью *интероперабельные* языки, поэтому одна и та же программа может быть частично написана на Котлине, частично на Java. Программы на Котлине могут использовать все имеющиеся Java-библиотеки, и наоборот. На данный момент программы на Котлине пишут сотни тысяч программистов, основная ниша его промышленного применения — мобильные программы под платформу Android и, в несколько меньшей степени, web-разработка.

В ходе изучения Котлина мы изучим также многие элементы стандартной библиотеки Java, а понимание работы программ на Котлине во многом упростит понимание работы Java-программ.

Что требуется для начала

Самый простой способ начать программировать на Котлине — зайти на сайт http://try.kotlinlang.org. Имеющаяся там "песочница" позволяет писать программы прямо в браузере, с возможностью выполнять и сохранять свои программы и проходить обучающие курсы.

Масштабы песочницы, однако, достаточны только для небольших программ, а более-менее серьёзные программы, как правило, разрабатываются в интегрированной среде (IDE). Разработка под платформу Java в любом случае требует установки пакета JDK, который необходимо скачать с сайта компании Oracle. Первое время вам потребуется Java Platform, Standard Edition, рекомендуется использовать 8-ю или 11-ю редакции. На сентябрь 2019 года последние версии — это Java SE 8u221 и Java SE 11.0.4 соответственно. 12-я редакция Java SE является экспериментальной и её использование не рекомендуется.

В качестве интегрированной среды разработки рекомендую установить IntelliJ IDEA Community Edition, её следует брать отсюда. Community Edition является полностью бесплатной, базовая версия обеспечивает поддержку программирования на Java, Kotlin, Scala, Groovy, поддержку систем контроля версий Git, Mercurial, SVN, интеграцию с системами сборки Maven и Gradle.

Для интеграции IDEA с системой контроля версий Git необходимо установить один из клиентов Git. Таких клиентов существует много; "родной" Git клиент можно скачать здесь. Имейте в виду, что в IDEA интегрирован собственный Git-плагин, уже имеющий графический интерфейс, поэтому скачивать и устанавливать клиенты Git с графическим интерфейсом (GUI Clients) необязательно.

Учебный проект

В ходе обучения мы будем активно использовать проект "Котлин как первый язык программирования", содержащий текст данного пособия и около сотни различных задач на языке Kotlin. Оригинальный код данного проекта доступен по адресу https://github.com/Kotlin-Polytech/KotlinAsFirst2019 на сайте GitHub, который является специализированным хранилищем программных кодов и основан на системе контроля версий Git. Для того,

чтобы начать работать с этим проектом, Вам необходимо выполнить следующие действия.

1. Зарегистрироваться на https://github.com/ (в случае, если у Вас еще нет GitHub аккаунта). Далее выбранное Вами имя будет обозначаться как <USERNAME>.
2. Создать специальную копию репозитория проекта — *форк*. Для этого достаточно зайти на страницу проекта https://github.com/Kotlin-Polytech/KotlinAsFirst2019 и нажать кнопку `Fork` в правом верхнем углу страницы. После этого Ваша персональная копия проекта станет доступна по адресу https://github.com/<USERNAME>/KotlinAsFirst2019, и всю работу по решению различных задач Вы должны выполнять именно с Вашей копией.
3. Для загрузки проекта в IntelliJ IDEA следует выполнить команду `Check out from Version Control → GitHub` из окна `Welcome to Intellij IDEA` (или `File → New → Project from Version Control → GitHub` из окна проекта), в появившемся окне ввести Git Repository URL https://github.com/<USERNAME>/KotlinAsFirst2019 и место на компьютере, куда будет скачан проект (Parent Directory).
4. Далее следуйте инструкциям среды для настройки проекта. Подробное руководство вы можете найти здесь.

Проект содержит задачи, разбитые на девять уроков (lesson). Тексты задач доступны через окно `Project` в IntelliJ IDEA (открывается комбинацией клавиш `Alt + 1`). В папках `src/lessonX`, где X — номер урока, находятся примеры решённых задач к данному уроку, тексты задач, которые необходимо решить, и готовые заглушки функций для написания решения. В папках test/lessonX находятся тестовые функции к задачам. Подробнее о задачах и тестах см. раздел 1 этого пособия.

Работа с проектом будет осуществляться в рамках системы контроля версий Git, а Ваши решения будут представлять собой коммиты в соответствующий репозиторий. Коммит — это законченное изменение текста проекта (например, решение одной или нескольких задач из какого-либо урока). Для более глубокого понимания того, что такое системы контроля версий, зачем они нужны и как с ними работать, рекомендуется пройти один или несколько обучающих уроков из Интернета. Несколько примеров подобных туториалов приведены ниже.

- https://githowto.com/ru/
- https://try.github.io/levels/1/challenges/1

В ходе освоения различных уроков вы можете обращаться за помощью к преподавателю или своим однокурсникам, но важно, чтобы *хотя бы одна* задача из каждого урока была решена вами самостоятельно. Имейте в виду, что задачи разбиты на пять уровней сложности, сложность задачи указана в комментарии к ней (тривиальная Trivial, простая Easy, средняя Normal, сложная Hard и очень сложная Impossible). Решайте в первую очередь те задачи, которые кажутся вам по силам, но иногда пытайтесь решать и более сложные. Если у вас нет идей, как решить какую-либо сложную задачу, — попросите преподавателя о подсказке. Оценка по итогам курса будет зависеть от сложности решённых вами задач из различных уроков.

Законченное решение задачи (или нескольких задач из одного урока) следует фиксировать в виде коммита в системе контроля версий. Для этого необходимо:

1. Зайти в окно `Version Control` среды Intellij IDEA. Это делается из меню `View → Tool Windows`, либо с помощью комбинации клавиш `Alt + 9`.
2. В нём щелчком мыши выбрать вкладку `Local Changes` (локальные изменения). Убедитесь, что файлы, которые вы меняли, присутствуют в этом окне.
3. В контекстном меню выберите команду `Commit Changes`. В появившемся окне введите осмысленный комментарий к вашему изменению (например, "Решена задача такая-то"), откройте выпадающее меню справа от кнопки `Commit` и в нём выберете команду `Commit and Push`.
4. При появлении соответствующих окон введите своё имя и e-mail для идентификации автора коммита (эти поля заполняются один раз), а также логин и пароль для Вашего аккаунта на GitHub.

Система Kotoed

Проверка зафиксированного решения какого-либо урока или его части будет осуществляться через систему Kotlin Online Education (aka `Kotoed`), расположенную по адресу: https://kotoed.spbstu.ru. Для работы с ней следует выполнить следующие действия:

1. Зарегистрироваться в системе, указав свой никнейм, почту и пароль.
    - Для упрощения входа в систему можно связать свой аккаут с одним или несколькими OAuth провайдерами при помощи соответствующих ссылок на странице логина.
2. В профиле указать ваше имя (First Name), фамилию (Second Name) и номер группы (Group).
3. На странице нашего курса KotlinAsFirst-2019 создать проект (`Create project`), связанный с вашим репозиторием на GitHub.

В данном проекте вы будете создавать запросы на проверку (submissions), в рамках которых будет осуществляться оценка как корректности вашего решения, так и качества вашего кода. Для создания запроса на проверку вам следует зайти в системе Kotoed на страницу вашего проекта и нажать `Submit`, при этом будет автоматически создан запрос на проверку *последней* ревизии (версии) вашего репозитория. Если вы хотите, чтобы проверялась какая-то конкретная ревизия, это можно сделать через `Specify revision` в выпадающем меню кнопки `Submit`.

Созданный запрос будет автоматически проверен через какое-то время, о чем вам придет соответствующее уведомление. После этого на странице с результатами запроса (`Results`) вы сможете увидеть следующую информацию:

- Какие задачи были решены верно
- Какие задачи были решены неполностью или неправильно, с указанием непрошедших тестов
- Статистику решения заданий по всем урокам
- Ошибки сборки проекта (если такие имеются)

- Ошибки запроса на проверку (если такие имеются)

Кроме того, на странице `Review` вы можете как задать преподавателю вопрос по какому-либо заданию в виде комментария к интересующей вас строчке кода, так и увидеть вопросы и замечания преподавателей к вашему коду. Для того, чтобы начать или продолжить обсуждение, следует нажать на карандаш рядом с интересующей вас строкой или на отметку об имеющихся комментариях. Для того, чтобы добавить новый комментарий к уже имеющимся, можно воспользоваться формой добавления комментария.

Процесс внесения исправлений в уже созданный запрос заключается в следующем.

1. Поправить найденные ошибки и замечания преподавателя в вашем репозитории, после чего зафиксировать их в виде одного или нескольких коммитов (`Commit and push`).
2. Зайти в текущий активный запрос на исправление и нажать кнопку `Resubmit`. При необходимости проверить конкретную ревизию можно воспользоваться выпадающим меню `Specify revision`.

После этого будет создан новый зависимый запрос на исправление, в который автоматически перенесутся все комментарии из его родителя. Он точно так же, как и обычный запрос на исправление, будет проверен, о чем вам придет соответствующее уведомление.

Как только ваш запрос будет удовлетворять требованиям преподавателей, он будет закрыт, после чего вы можете приступать к решению следующих уроков.

Полезные советы

Несмотря на то, что у многих из вас будет желание двигаться вперед как можно скорее, мы убедительно просим вас придерживаться следующих трех правил.

- В том и только том случае, если все задачи, которые Вы хотели решить, успешно проверены и ваш запрос закрыт преподавателем, Вы можете приступать к следующим задачам.
- Если часть задач решена неправильно, постарайтесь исправить возможные ошибки при помощи предоставленных Вам тестов.
- В случае, если Вы уверены в правильности решения или не можете понять, где Вы ошиблись при решении, можете обратиться к преподавателю.

В случае обнаружения ошибок и недоработок в системе Kotoed можно и нужно сообщить о них преподавателям, которые постараются исправить их как можно скорее.

Вопросы

В ходе изучения нового языка у вас, конечно, будут возникать вопросы, не стесняйтесь их задавать. Помимо обращения к вашим однокурсникам и преподавателям, у вас есть следующие возможности:

- посмотреть "часто задаваемые вопросы" далее по тексту
- поискать ответ на вопрос с помощью поисковой системы в Интернете
- почитать разнообразную информацию о Котлине в его документации
- задать нам вопрос в Kotlin Slack (получить приглашение можно здесь) в канале russian-kotlinasfirst
- воспользоваться другими ресурсами для общения

Kotlin Slack — это система общения, созданная специально для программистов на Котлине. Система состоит из множества каналов, посвящённых разным аспектам программирования на Котлине — в большинстве из них общение идёт на английском языке. Нашему курсу посвящён канал russian-kotlinasfirst, и там вы сможете задать любые вопросы по этому курсу на русском языке. В качестве других важных каналов назову general — канал с общими обсуждениями, касающимися Котлина, и russian — общий канал для русскоязычных Котлин-программистов.

Часто задаваемые вопросы (F.A.Q.)

- Что делать, если при открытии файла расширением .kt из учебного проекта (например, Simple.kt) вы видите сообщение над ним `Project SDK is not defined`?

Нажмите на ссылку `Setup SDK` в правой части сообщения. Выберете JDK 1.8 для работы с проектом в появившемся окне. Если список JDK в окне пуст или не содержит JDK 1.8, следует нажать на клавишу `Configure`, затем зелёный плюс в верхнем левом углу и зарегистрировать установленную на Вашем компьютере JDK 1.8 в Intellij IDEA. Если Вы забыли установить JDK, это следует сделать, предварительно скачав её с сайта Oracle.

- Что делать, если отсутствует зелёный треугольник напротив функции `main` и тестовых функций?

Откройте окно Maven Projects на панели в правой части окна IDEA (если вы не видите там такой надписи, откройте его через меню — View > Tool Windows > Maven Projects) и нажмите в нём на кнопку с изображением двух стрелок в круге. Дождитесь окончания импортирования Maven-проекта (наблюдайте за надписями в нижней части окна IDEA), после чего зелёные треугольники должны появиться. Проверьте также отсутствие надписи `Project SDK is not defined` в верхней части окна (см. вопрос выше).

Если вам не удаётся открыть окно Maven Projects, попробуйте выйти из Intellij IDEA и войти в неё заново.