Высшая школа информационных технологий и информационных систем

ЛЕТНЯЯ
ИТИС-ШКОЛА

Барух САДОГУРСКИЙ,

Developer Advocate в JFrog (Израиль)

Самый остроумный и прагматичный тренер, по мнению студентов-практиков ИТИС. Убедитесь сами на Youtube

В прошлом: BMC Software Inc и AlphaCSP. Изучал Information Technology Management в Политехнический институт Нью-Йоркского университета. Еще ранее: Champlain College и Bosmat - Junior Technical College.

Живет в Tenuvot.

приглашаем на мастер-классы

VIP-тренера Баруха САДОГУРСКОГО

для школьников и абитуриентов:

- *Basic Java - Введение в Java*
- *Design Patterns - Шаблоны проектирования*

(Казань, 1-4 июля)

для профессионалов:

- *Maven Gradle*
- *Groovy*

(Казань, 25-29 августа)

Высшая школа информационных технологий и информационных систем

Design Patterns, Basic Java (1-4 июля),
Maven Gradle / Groovy (25-29 августа)

ЛЕТНЯЯ ИТИС-ШКОЛА: 221-34-33 (доб. 12)

Введение в Java,
или 3 миллиарда устройств ждут твоих приложений

(Казань, 1-4 июля)

***Целевая аудитория*:**

Курс будет полезен тем, кто планирует всерьез заняться изучением программирования вообще, и программированием на Java в частности.

Что вы узнаете на тренинге:

Безусловно, выучить новый (и уж тем более, первый) язык программирования за несколько дней невозможно. Но можно "попробовать его на вкус", и можно облегчить дальнейшее углубленное обучение. Именно такие задачи ставит перед собой этот тренинг.

Мы пройдем по основным концепциям, конструкциям и особенностям языка, начиная от знакомства с виртуальной машиной, через основные понятия объектного ориентирования, синтаксис и структуру и заканчивая ознакомлением с гигантской вселенной вспомогательных классов и библиотек, которые и превратили Java в безусловного лидера среди языков программирования

Шаблоны проектирования,
или как писать код, которым можно гордиться

(Казань, 1-4 июля)

Целевая аудитория:

Начинающие Java разработчики, желающие перейти на следующую стадию профессионализма, от "работает", к "правильно написано, расширяемо и легко поддерживается".

Что вы узнаете на тренинге:

После того, как вы выучили "слова и фразы" нового языка (будь то английский, или Java), пора переходить к следующей стадии
— учиться правильно его использовать. Шаблоны проектирования дают общую базу использования, стандартные схемы решения частых проблем и помогают лучше понять, как правильно "говорить" на языке.

Данный тренинг отличается от привычных всем тренингов по шаблонам проектирования, он основан на реальных примерах, на чужой боли и суровом опыте. Здесь вы не столкнетесь с нудной теорией, которая редко применяется на практике.
Вы будете решать реальные задачи, с которыми сталкивается почти каждый программист. Будут обсуждаться как правильные, так и неправильные шаблоны, названия которых будут озвучены только после того, как качественный код будет написан.

С этого тренинга вы унесете чужой жизненный опыт и собственноручно написанный красивый код, которым можно гордиться.

Программы

Basic Java (Введение в Java)

- Введение в Java. История создания. Преимущества.
- Словарь Java
- Примитивные и объектные типы в Java.
- Операторы.
- Управляющие структуры
- Процедурное и объектно-ориентированное программирование. Инкапсуляция.
- Объекты и объектные переменные. Сборка мусора.
- Пакеты. Уровни видимости классов. Импорт классов.
- Введение в IntelliJ IDEA.
- Документирование кода в Java. Javadoc.
- Построение иерархии классов. Диаграммы классов UML.
- Методы. Модификаторы. Передача примитивных типов в методы.
- Локальные и глобальные переменные.
- Модификаторы доступа и правила видимости. Зарезервированное слово this.
- Передача ссылочных типов в методы.
- Наследование. Суперклассы и подклассы. Переопределение методов.
- Наследование и правила видимости. Зарезервированное слово super.
- Статическое и динамическое связывание методов. Полиморфизм.
- Базовый класс Object.
- Конструкторы. Зарезервированные слова super и this.
- Блоки инициализации.
- Удаление неиспользуемых объектов и метод finalize.
- Проблема деструкторов для сложно устроенных объектов.
- Перегрузка методов.
- Правила совместимости ссылочных типов. Приведение и проверка типов.
- Рефакторинг.
- Реверс-инжиниринг.
- Проблемы множественного наследования классов. Интерфейсы.
- Отличия интерфейсов от классов. Наследование интерфейсов.
- Пример использования интерфейсов.
- Композиция как альтернатива множественному наследованию.
- Вложенные (Nested) классы.
- Внутренние (Inner) классы.
- Локальные (local) классы.
- Анонимные (безымянные) классы и обработчики событий.
- Обзор основных пакетов Java.
- Массивы в Java.
- Классы String, StringBuffer, StringBuilder, StringTokenizer.
- Классы ввода-вывода. Файловый ввод-вывод.
- Интерфейс Collections.
- Система исключения в Java
- Assertions или проверка утверждений
- Порядок действий VM при загрузке класса
- Способы загрузки классов.
- Создание собственного загрузчика классов
- Пример создания загрузчика зашифрованных классов
- Загрузка классов методом forName (…).
- Создание объектов класса Class.
- Исследование объекта.
- Класс java.lang.reflect.Method.
- Выгрузка и Перезагрузка Классов. Пример приложения.
- Понимание настраиваемых классов или Generics
- Написание кода, поддерживающего Generics
- Понимание аннотаций
- Создание собственных аннотаций
- Обработка аннотаций
- Что такое Сериализация и где применяется.
- Как сериализация может использоваться в RMI.

- Автоматическая сериализация. Интерфейс Serilizable .Пример.
- Сериализация с применением интерфейса Externalizable. Пример.
- Восстановления объекта и Reflection. Пример приложения.
- Управление сериализацией посредством transient. Пример приложения.
- Альтернатива Externalizable – методы writeObject() и readObject().
- Сериализация static членов классов.
- Глубокое клонирование объектов через сериализацию. Пример с Reflection.
- Введение в многопоточную архитектуру
- Базовые классы для работы с потоками – класс Thread, интерфейс Runnable.
- Синхронизация, Блокировки.
- Введение в сетевое программирование, сокеты
- Классы Soket и ServerSocket, написание простого сетевого клиента и сервера.

Design Patterns (Шаблоны проектирования)

- Принципы правильного кода
- Java оптимизации + правильный код = хорошая производительность
- Антипаттерны
- Switch as anti-pattern
- Garbage collection + immutable objects
- Builder
- Strategy
- Adapter
- Template method
- Iterator
- Как написать хороший framework
- Factory
- Singleton
- Composite
- Dependency injection
- Inversion of control
- Callback method (Closures)
- Proxy
- Decorator
- Observer
- Command
- Chain of responsibility

Современные системы сборки и управления зависимостями для Java разработчиков

(25-29 августа)

***Целевая аудитория*:**

Java разработчики, которые хотели бы получить теоретические и практические навыки работы с наиболее популярной сегодня системой сборки Maven и с набирающей популярность системой Gradle, а также разобраться в принципах и деталях их устройства.

Что вы узнаете на тренинге:

Пользовались ли вы средой разработки для сборки проектов, писали ли скрипты или xml-файлы на Ant, вы безусловно чувствовали

– должен быть способ лучше.
И он есть. Использование современной системы сборки, декларативной, стандартной, и со встроенным управлением зависимостями, приводит автоматизацию сборки проекта на совершенно новый уровень, на котором вы можете заниматься тем, за что вам платят деньги и что вы любите делать

— писать продукт, а не воевать со сборкой.
На этом тренинге вам будут представлены два безусловных лидера среди систем сборки

— Apache Maven, являющийся сегодня стандартом де-факто, и Gradle, оспаривающий этот титул. Знание обоих позволит вам легко влиться в подавляющее большинство существующих команд разработки, а так же выбрать достойную систему сборки для вашего нового проекта.

Groovy
– младший брат Java на веществах
– дольше, мощнее, сильнее!

(25-29 августа)

Целевая аудитория:

Java разработчики, которые хотели бы познакомиться с идеальным Java компаньоном - динамическим, опционально-типизированым и функциональным, полностью объектно-ориентированным языком программирования с Java-подобным синтаксисом.
Если вы любите Java, но всегда чувствовали, что компилятор издевается над вами, говоря, где у вас не хватает точки-с-запятой, но настаивая, чтобы вы ручками туда её добавили - этот тренинг для вас.

Что вы узнаете на тренинге:

Казалось бы, зачем нам ещё один язык на JVM? Java мощна, объектно-ориентирована и богата фреймворками выше всяких границ. Куда уж больше?

Но будем честны друг с другом: иногда всем нам хочется видеть в Java немножко меньше бойлерплейта, немножко больше динамизма и функциональщины. Но при этом очень не хочется учить новый язык и отказываться от всех плюшек Java, включая всю экосистему.

Groovy — он как Java, только лаконичней, динамичней и намного функциональней (во всех смыслах слова). Опционально-типизированный, объектно-ориентированный язык программирования на JVM, с Java-подобным синтаксисом и двухсторонней совместимостью как с самой Java, так и с любым Java-фреймворком и любой Java-библиотекой, с поддержкой метапрограммирования (как на уровне компилятора, так и во время исполнения) и с полной поддержкой лямбда-выражений.

Впечатляет? И не зря. Благодаря всему вышеперечисленному, Groovy является наиболее популярным языком на JVM после Java, имеет миллионы скачиваний по всему миру, на нем написаны супер-популярные фреймворки для параллельных вычислений, RAD-разработки, тестирования и сборки.

На этом тренинге вы познакомитесь с Groovy, почувствуете как просто перейти с Java на Groovy (и обратно), узнаете о различиях между двумя языками, о плюсах и минусах Groovy по сравнению с Java и поймёте, когда стоит использовать каждый из этих языков.

Вы нырнёте с головой в различия синтаксиса для работы с классами, строками и коллекциями, в динамическое, функциональное и мета-программирование на Groovy и познакомитесь с новыми классами из Groovy SDK, которые облегчат вам повседневные задачи разработки.

Вы научитесь работать с доступными в Groovy фреймворками для многопоточного программирования, доступа к базам данных и обработки XML и JSON, а так же узнаете о способах «протаскивания» Groovy в свою организацию, используя «черные дыры» инструментов тестов и сборки.

Программы

Maven & Gradle

- Что такое система сборки проекта и зачем она нужена?
- Императивные vs Декларативные системы сборки

***Maven*:**

- Веедение в Maven
- Установка Maven
- Архетипы
- Структур POM-файла
- Управление зависимостями
- Свойства
- Жизненный цикл проекта
- Часто используемые плагины
- Интеграция с IntelliJ IDEA
- Использование профилей
- Создание плагинов

Gradle:

- Введение в Gradle
- Установка и настройка Gradle
- Билд скрипты Gradle
- Жизненный цикл сборки
- Задачи
- Работа с логами
- Плагины
- Интеграция с Ant
- Управление зависимостями
- Расширение модели
- Задачи ввода и вывода
- Java плагин
- Многомодульная сборка
- Процесс сборки
- Итоговое сравнение систем сборки

Введение в Groovy

- Груви для тебя
- Увертюра - знакомство в Груви
- Простые типы данных
- Коллективные типы данных
- Работа с замыканиями
- Управляющие структуры
- ООП в Груви
- Динамическое программирование в Груви
- Мета-программирование на уровне компилятора и AST трансформации
- Груви как статический язык
- Билдеры
- GDK
- Работа с базами данных
- Работа с XML и JSON
- Работа с Web services
- Встраиваемый Груви
- Тестирование в Груви
- Многопоточное программирование с помощью GPars
- Domain Specific Languages (DSLs)
- Экосистема Груви

Стоимость участия – БЕСПЛАТНО
Количество билетов ограничено

Регистрация обязательна:

для школьников и абитуриентов,
http://itiskpfu.timepad.ru/event/127113
для профессионалов,
http://itiskpfu.timepad.ru/event/127116

Ждем Вас этим летом: ул. Нужина 1/37

Контакты организаторов:

420008 Россия, Республика Татарстан, г. Казань, ул. Нужина 1/37
Тел.:(843) 221-34-33 (доб. 12),
it@kpfu.ru www.it.kfu.ru

Высшая школа ИТИС создана в 2011 году совместными усилиями Министерства информатизации и связи РТ, Казанского федерального университета, мировых брендов IBM, Microsoft, HP, Cisco, Oracle, представителей крупнейших IT-компаний региона.

Особенностью ИТИС КФУ является система грантов для абитуриентов и студентов и промышленно-прикладная направленность исследований и обучения.

В промышленных лабораториях, созданных совместно с IT-бизнесом, студенты с 1–го курса участвуют в реальных проектах, разрабатывают программное обеспечение, веб-сервисы и мобильные приложения, решают исследовательские задачи для нужд предприятий индустрии информационных и компьютерных технологий.