Eğitim Kataloğu

2019

http://academy.eteration.com

academy@eteration.com

eteration bilişim çözümleri ve ticaret anonim şirketi itü arı-3 teknokent B202 maslak Istanbul 34469 türkiye tel: +90 212 328 0825 fax: +90 212 328 0521

1. ETERATION HAKKINDA 4

2. EĞİTİM HİZMETLERİMİZ 6

3. EĞİTİMLERİMİZ 8

3.1 Cloud Technologies 8

3.1.1 Building Applications with Microservices , Docker and SpringBoot 8

3.1.2 Cloud Native Patterns 9

3.1.3 Docker Fundamentals 12

3.1.4 Docker for Enterprise Operations 13

3.1.5 Docker for Enterprise Developers 14

3.1.6 Docker Fundamentals and Enterprise Operations 15

3.1.7 Kubernetes & Docker Fundamentals 17

3.1.8 MicroServices with Kubernetes 19

3.1.9 Kubernetes for Enterprise Operations 21

3.2 Web Technologies 23

3.2.1 Developing Client-Site Web Applications with ReactJS 23

3.2.2 Next-generation Web Applications with full stack JavaScript 24

3.3 Spring Technologies 26

3.3.1 Developing Enterprise Applications using Spring Framework and JPA 26

3.3.2 Spring Boot 30

3.4 Java Technologies 31

3.4.1 Developing Object-Oriented Programs in Java 31

3.4.2 Effective JAVA Programming Using Design Patterns 34

3.4.3 Java New Features 38

3.4.4 Effective Agile Java Development 40

3.4.5 Building Secure Applications with Java and JavaEE 44

3.4.6 Effective Testing for Java developers 46

3.4.7 Writing Elegant and Readable Code 49

3.5 Mobile Technologies 51

3.5.1 Developing Mobile Applications with ReactNative 51

3.6 Executive Training **54**

3.6.1 Technology Trends for Executives 54

3.7 Analysis and Design **55**

3.7.1 Object Oriented Analysis and Design using UML 55

3.8 Project Management **56**

3.8.1 Agile Methodology & Scrum 56

3.8.2 Agile Methodology & Scrum& Agile Testing 58

3.9 Software Project Management **62**

3.9.1 Apache Maven 62

3.9.2 Gradle 64

3.10 SQL **66**

3.10.1 PL/SQL Programming 66

1. ETERATION HAKKINDA

2002 yılında kurulan Eteration, Türkiye, Avrupa ve Asya'da yer alan çeşitli sektörlere üstün yazılım, danışmanlık ve eğitim hizmetleri sunmayı hedefleyen bir Türk yazılım ve AR-GE kuruluşudur. Kurumsal Java, Mikroservisler, Bulut Teknolojileri, Dijital Dönüşüm, Hizmet Tabanlı Mimari (SOA), Analiz, Web ve Mobil Teknolojileri konularında kapsamlı, disiplinli ve metodolojik yaklaşımı ile öne çıkarak tercih nedeni olmaktadır.

Kurulduğu günden beri istikrarlı bir büyüme gösteren Eteration üç kişi ile başladığı serüvenine bugün 50'nin üzerinde mühendisle devam etmektedir. Müşterileri arasında Türkiye'nin üç büyük GSM operatörü, büyük finans kuruluşları ve bazı kamu kuruluşları yer almaktadır.

Eteration bir AR-GE şirketi kimliğiyle İTÜ ARI Teknokent'te faaliyetlerine devam etmektedir.

Eteration;

- Uluslararası alanda IT uzmanlığına sahip tecrübeli bir ekipten oluşmaktadır.
- Mikroservisler mimarisi ile framework ve uygulamaların tasarlanması ve geliştirmesi.
- Servis Tabanlı çözümlerin mimari çalışmalarını ve üretimini gerçekleştirir.
- Karmaşık Web Portal, içerik çözümleri ve mobil uygulamalar tasarlar ve üretir.
- En yeni Java, Bulut ve Kurumsal uygulamaların entegrasyonlarını gerçekleştirir.
- Yazılım Proje Yönetimi, Mimari ve Yazılım Geliştirme süreçleri ve uygulamaları konularında eğitim hizmetleri sunar.
- Yazılım Geliştirme süreçlerini hızlandıracak ve katkıda bulunacak framework ve araçlar geliştirmektedir.

Hizmetlerimiz;

Eteration Profesyonel Hizmetler Grubu (PHG), müşterilerine temel iş süreçlerini Internet teknolojileri ile bütünleştirecek çözümler sağlamaktır. Çözümlerimizi mevcut alt yapı, teknoloji ve yatırımlar ile bütünleşik olarak oluşturmak en önemli önceliklerimizdendir.

Eteration PHG, bütünleşik Internet çözümlerini yaratmak için gerekli teknolojileri sağlar ve bu teknolojileri kurumunuzun uygulama geliştirme çalışmalarına temel oluşturacak alt yapılar halinde şekillendirir. Bu çalışmaları sizlerin uygulama geliştirme ekipleri ile birlikte yaparak, uygulama geliştirme süreçlerinin kurumunuz yapısına göre şekillendirilmesini sağlar. Tüm bu çalışmalar için gerekli alt yapı ve ileri düzey eğitimleri oluşturur ve kurumunuz, kendi içerisinde yeterli ve etkin bir ileri teknoloji grubu oluşturmasına yardımcı olur.

Hizmetlerimizi aşağıdaki gruplar içerisinde sunmaktayız:

- Uygulama Geliştirme Hizmetleri
- Dijital Dönüşüm Hizmetleri
- Eğitim Hizmetleri
- Immersion (Hızlandırılmış Teknoloji Transferi)
- Danışmanlık Hizmetleri
- Yazılım AR-GE Çalışmaları

2. EĞİTİM HİZMETLERİMİZ

Eteration eğitimleri yazılım ve uygulama geliştirmeye yönelik süreçler, analiz, tasarım ve programlamayı içine alan konuları içermektedir

Uygulama geliştiricilerin proje öncesindeki teknik alt yapılarının oluşturulmasına yönelik olarak hazırlanan eğitimler, katılımcılara başarılı bir uygulama geliştirmek için gerekli olan teorik ve pratik donanımları kazandırmayı hedeflemektedir. Eteration eğitimleri, uygulama geliştirme ve proje tecrübesine sahip eğitmenler tarafından verilmesiyle klasik anlamdaki sınıf eğitimlerinden ayrılmaktadır.

Sizle yapılan kapsam ve içerik belirleme çalışmalarıyla Eteration eğitimleri proje ihtiyaçlarına cevap verecek şekilde düzenlenebilmektedir. Geniş katılımlı eğitim programlarında, katılımcıların eğitim merkezine seyahatlerini ortadan kaldırmak ve de yapılacak proje ile ilgili ortama yakın olabilmek için eğitimleri Müşterinin kendi yerinde yapabilmekteyiz.

Eğitimlerimiz aşağıda belirtilen formatlarda verilmektedir:

- **Firmanın Kendi Yerinde Eğitim (Onsite)**

Proje ihtiyaçlarına göre düzenlenmiş, sizin göstereceğiniz lokasyonda verilen kurslar

- **Açık Kurslar (Public)** Eteration Eğitim Merkezinde verilen sınıf eğitimleri

Uygulama geliştiricilerin proje öncesindeki teknik alt yapılarının oluşturulmasına yönelik olarak hazırlanan eğitimler, katılımcılara başarılı bir uygulama geliştirmek için gerekli olan teorik ve pratik donanımları kazandırmayı hedeflemektedir. Eğitimler, uygulama geliştirme ve proje tecrübesine sahip eğitmenler tarafından verilmesiyle klasik anlamdaki sınıf eğitimlerinden ayrılmaktadır.

Eğitim araçları, materyalleri ve ortamı

Eteration, uygun niteliklerdeki eğitmen ya da eğitmenleri, tüm gerekli eğitim içeriklerini ve materyallerini eğitimin hedeflenen kalitede olmasını sağlamaktadır (eğitim kitapları, CD ler, gerekli yazılımlar, vb.).

Firmanın kendi yerinde düzenlenecek eğitimler için ortamın, çalışma alanlarının, araçların sağlanmasında firma kaynakları kullanılır (Her bir katılımcı için uygun özelliklere sahip bilgisayar, sunum cihazı, günlük çalışma alanından ayrı bir mekân, vb.).

Önerilen Donanım ve Yazılım Özellikleri

Eğitim çalışmasının firma ortamında gerçekleştirilmesi ve firmanın bilgisayarlarının kullanılması durumunda, kullanılacak bilgisayar özelliklerinin aşağıdaki gibi olmasını tavsiye ediyoruz:

- En az 4GB veya üzeri RAM bulunması,
- USB veya DVD benzeri harici saklama birimlerini desteklemesi,
- 20 GB boş alana sahip olması,
- Tercihen i5 veya üstü işlemcili olması,
- Eclipse ve JDK8 yüklü olması.

Talebiniz durumunda yukarıdaki özellikleri sahip Eteration eğitim bilgisayarları eğitim yerine getirilebilecektir.

3. EĞİTİMLERİMİZ

Kategorilere göre egitimlerimizin listesi aşağıdaki gibidir.

3.1 Cloud Technologies

3.1.1 Building Applications with Microservices , Docker and SpringBoot

Code

A-MSRVS

Overview

Microservice architecture, is a distinctive method of developing software systems that has grown in popularity in recent years. In fact, even though there isn't a whole lot out there on what it is and how to do it, for many developers it has become a preferred way of creating enterprise applications. Thanks to its scalability, this architectural method is considered particularly ideal when you have to enable support for a range of platforms and devices—spanning web, mobile, Internet of Things, and wearables—or simply when you're not sure what kind of devices you'll need to support in an increasingly cloudy future.

The big idea behind microservices is to architect large, complex and long-lived applications as a set of cohesive services that evolve over time.

Topics Covered

- What is a microservice?
- Microservice architecture characteristics
- Spring & Microservices
- Cloud Native Architecture
- Microservice with Spring Boot
- Cloud and Microservices
- IaaS & Microservices
- Best practices for microservice applications
- Spring Cloud & Microservices
- Spring Cloud
- Spring Boot
- Docker
    - Basic Docker Usage
    - Docker installation
    - Docker & Software Installation
    - Docker & Volumes
    - Docker & Network
    - Docker & Infrastructure as CODE!
    - Packaging Software for Distribution
- Decomposition & Microservices
- Spring Boot Microservices

- Spring Rest Services
- Microservices & Devops
- Spring Boot Actuator
- Spring Cloud Config
- Service Discovery with Spring Netflix Eureka
- Client Resiliency patterns with Spring Cloud Netflix Hystrix
- Service Routing with Spring Cloud and Zuul

Audience

Application Architects, Software Developers, Designers.

Duration

3 days

Format

Instructor Lead

Prerequisites

Core Java Syntax - (D-EJAVA-301-001) Effective JAVA Programming, JavaEE, Spring Framework

3.1.2 Cloud Native Patterns

Code

A-CNP

Overview

Companies are seeking the ability to continuously evolve new and existing value streams while simultaneously seeking increased reliability from all of their systems. The conversations within Continuous Delivery and DevOps communities, coupled with a deep understanding of the unique characteristics of cloud infrastructure platforms, have resulted in an evolving architectural pattern language geared toward helping companies achieve these two conflicting goals.

Designed for software architects and senior developers working on medium-to-large scale enterprise systems, this two-day, hands-on course will introduce you to the cloud native architectural pattern language and give you practice applying it. By the end of this course, you'll be able to articulate the high-level narrative of cloud native architecture and why it is so important to your company

Topics Covered

Part 1

1. Defining Cloud Native
   - Todays application requirements
   - Defining cloud native
   - what is not cloud native
2. Running CN Apps in prod
   - The obstacles (snowflakes, Risky deployments, etc)
   - continuous delivery, safe deployments, repeatability
3. The Platform for cn software
   - cn platform evolution
   - Core tenets of the cn platform
   - Capabilities

Part 2

1. Event Driven microservices: Its not just request/response
   - Reintroducing event-driven computing
   - Introducing Command Query Responsibility Segregation
2. App Redundancy: Scale out and statelessness
   - cn apps have many instances deployed
   - stateful apps in the cloud(decomposing the monolith and binding to the db)
   - Http & sticky sessions
   - Stateful services and stateless apps(making apps stateful)
3. App Config: Not just environment variables
   - dynamic scaling
   - increasing & decreasing # of app instances
   - infrastructure changes causing configuration changes
   - updating app configs with zero downtime
   - apps configuration layer
   - injecting system/environment values (In action ENV variables for configs)
   - Introducing the config server (In action)
4. The App lifecycle: accounting for constant change
   - Single app lifecycle, multiple instance lifecycles
   - blue/green deployments, rolling updates, parallel deployments
   - Coordinating across different app lifecycles
   - in action: credential rotation and app lifecycle
   - dealing with ephemeral rutime environments
   - app state → in action: health endpoints and probes
   - serverless
5. Accessing apps: Services, routing, and service discovery
   - The service abstraction
   - dynamic routing
   - server-side load balancing / client side load balancing/route freshness
   - Service discovery
   - service discovery with client side load balancing
   - service discovery with kubernetes
   - in action

6. Interaction Redundancy: retries & other control loops
    - request retries
    - in action: simple retries
    - what could go wrong ?
    - in action: retry storm
    - avoiding retry storms
    - in action:
    - fallback logic (in action: implementation)
    - control loops
7. Fronting services: Circuit breakers & api gateways
    - circuit breakers
    - implementing circuit breakers
    - api gateways
    - Control loops
8. Troubleshooting: finding needle in the haystack
    - application logging
    - Pulling & pushing metrics from cn apps
    - the service mesh
9. Cloud Native data: breaking the data monolith
    - Cache
    - request/response to event-driven
    - in action : event-driven microservices
    - event sourcing
10. Cheatsheet
    - request/response
    - event-driven
    - CQRS(Command query responsibility segregation)
    - Multiple service instances
    - horizontal scaling
    - stateless services
    - stateful services
    - app configuration through environmental variables
    - config server
    - configuration as code
    - Zero-downtime upgrades
    - rolling updates
    - blue/green upgrades
    - application health checks
    - Liveness probes
    - server-side load balancing
    - client side load balancing
    - service discovery
    - retry
    - safe-service
    - idempotent service
    - fallbacks
    - circuit-breaker
    - api gateways
    - sidecars
    - service mesh

- distributed tracing
- event sourcing

Audience

Software architects, DevOps Engineers, Senior software developers, System designers

Duration

2 days

Format

Instructor Lead

Prerequisites

Basic knowledge of microservices architecture

3.1.3 Docker Fundamentals

The Docker Fundamentals training course features the foundational concepts and practices of containerization on a single Docker node. The course offers learners the opportunity to assimilate basic container orchestration and how to scale Docker across multiple nodes in a simple swarm cluster. This course provides essential foundational knowledge for subsequent Docker courses.

By the end of the course successful learners will be able to:

- Understand the foundations of containerization on a single Docker node
- Create an image using Dockerfile best practices
- Use volumes in the application development process
- Apply concepts of the Docker networking model
- Understand the goal of services as a method of scaling containers
- Utilize two different orchestrators (Swarm and Kubernetes) to deploy a single application across multiple machines
- Create a secret and understand its accessibility capabilities

Description

- Introducing Docker
- Containerization Fundamentals
- Creating Images
- Docker Volumes
- Docker Networking Basics
- Introduction to Docker Compose
- Introduction to Swarm Mode
- Introduction to Kubernetes

- Secrets
- Fundamentals Signature Assignment

Audience

Developers, operators, and architects desiring a strong foundation in Docker technologies and an introductory hands-on experience building, shipping, and running Docker containers.

Duration

2 days

Format

Instructor Lead

3.1.4 Docker for Enterprise Operations

As the follow-on to the Docker Fundamentals course, Docker for Enterprise Operations is a role-based course designed for an organization's Development and DevOps teams to accelerate their Docker journey in the enterprise. The course covers in-depth core advanced features of Docker EE and best practices to apply these features at scale with enterprise workloads. It is highly recommended to complete the Docker Fundamentals course as a pre-requisite

By the end of the course successful learners will be able to

- Identify the key features of UCP and DTR
- Build a complete, basic software supply chain using UCP and DTR that includes CI/CD, content trust, and image scanning
- Describe the methodological differences between managing containers and managing virtual machines
- Deploy applications on Swarm or Kubernetes orchestrators via UCP

Description

- Introduction to Docker Enterprise Edition
- Universal Control Plane
- User Management and Access Control
- UCP Orchestration
- Service Mesh
- Logging
- Application Health and Readiness Checks
- Docker Trusted Registry
- DTR Organizations and Teams
- Content Trust
- Image Security Scanning
- Repository Automation
- Image Caching
- Operations Signature Assessment

DURATION

2 Days

PREREQUISITES

Completion of the Docker Fundamentals course or its equivalent.

3.1.5 Docker for Enterprise Developers

As the follow-on to the Docker Fundamentals course, Docker for Enterprise Developers is a role-based course designed for an organization's Development and DevOps teams to accelerate their Docker journey in the enterprise. The course covers best practices to containerize and modernize legacy applications or build containerized applications from scratch that are secure, robust, highly available, resilient and self-healing. It is highly recommended to complete the Docker Fundamentals course as a pre-requisite.

By the end of the course successful learners will be able to:

- Describe the essential patterns used in a highly distributed EE application
- Understand how to configure EE applications for different environments without code changes
- Produce and containerize EE applications that are scalable, accessible, and fault-tolerant
- Apply different debugging and testing techniques to containerized EE applications
- Build and run the sample application on a local system using Kubernetes

Description

- Distributed Application Architecture
- Sample Application
- Edit and Continue
- Debugging
- Docker Compose
- Testing
- Service Discovery
- Health Checks
- Defensive Programming
- Logging and Error Handling
- Builder
- Docker Swarm and Kubernetes
- Secrets
- Configuration Management
- Development Pipeline Overview
- Universal Control Plane
- Context Based Routing
- Docker Trusted Registry
- Content Trust
- Image Security Scanning

- Repository Automation
- Tagging and Versioning Strategies
- Build Server

Audience

Software Engineers and DevOps professionals working in an Enterprise developing mission critical line of business applications

Duration

2 days

Format

Instructor Lead

Prerequisites

Completed Docker Fundamentals Course or equivalent

3.1.6 Docker Fundamentals and Enterprise Operations

The Docker Fundamentals + Enterprise Operations Bundle includes the full Docker for Enterprise Operations course as well as the prerequisite Docker Fundamentals course run back to back in a single intensive training experience. The Docker Fundamentals training course features the foundational concepts and practices of containerization on a single Docker node. The course offers learners the opportunity to assimilate basic container orchestration and how to scale Docker across multiple nodes in a simple swarm cluster. This course provides essential foundational knowledge for subsequent Docker courses. As the follow-on to the Docker Fundamentals course, Docker for Enterprise Operations is a role-based course is designed for Docker Operations teams to accelerate their Docker journey in the enterprise. The course covers in-depth core advanced features of Docker EE and best practices to apply these features at scale with enterprise workloads.

By the end of the course successful learners will be able to:

- Understand the foundations of containerization on a single Docker node
- Create an image using Dockerfile best practices
- Use volumes in the application development process
- Apply concepts of the Docker networking model
- Understand the goal of services as a method of scaling containers
- Utilize two different orchestrators (Swarm and Kubernetes) to deploy a single application across multiple machines
- Create a secret and understand its accessibility capabilities
- Identify the key features of UCP and DTR
- Build a complete, basic software supply chain using UCP and DTR that includes CI/CD, content trust, and image scanning.
- Describe the methodological differences between managing containers and managing virtual machines.
- Deploy applications on Swarm or Kubernetes orchestrators via UCP

Description

- Introducing Docker
- Containerization Fundamentals
- Creating Images
- Docker Volumes
- Docker Networking Basics
- Introduction to Docker Compose
- Introduction to Swarm Mode
- Introduction to Kubernetes
- Secrets
- Fundamentals Signature Assignment
- Introduction to Docker Enterprise Edition
- Universal Control Plane
- User Management and Access Control
- UCP Orchestration
- Service Mesh
- Logging
- Application Health and Readiness Checks
- Docker Trusted Registry
- DTR Organizations and Teams
- Content Trust
- Image Security Scanning
- Repository Automation
- Image Caching
- Operations Signature Assessment

Audience

IT professionals with an operations or system administration background desiring an intense and rapid onramp to Docker technologies

Duration

4 days

Format

Instructor Lead

3.1.7 Kubernetes & Docker Fundamentals

Code

A-KUB1

Overview

This course will teach you how to use the container management platform used by companies to manage their application infrastructure.

You'll learn:

- Kubernetes architecture
- Introduction to Docker
- Deployment
- How to access the cluster
- Secrets and ConfigMaps
- And much more!

OUTLINE

Kubernetes Fundamentals

1) Introduction
    a) Requirements
    b) References
    c) What is Kubernetes
    d) What problems solved with Kubernetes
    e) Kubernetes vs Docker Swarm
    f) Key concepts
2) Architecture
    a) Kubernetes Architecture & Components
    b) Master - Node Communication
3) Key Concepts
    a) Kubernetes Objects
    b) YAML Files
    c) Names
    d) Namespaces
    e) Pods
    f) Services
    g) Labels and Selectors
4) Workloads
    a) ReplicaSet
    b) Deployment

c) StatefulSet
d) DaemonSet
e) Job
f) CronJob

5) Services & Load Balancing
   a) Services
   b) Ingress
   c) Ingress Controllers
   d) Network Policies
   e) Adding Entries with HostAliases
6) Storage
   a) Volumes
   b) Persistent Volumes
   c) Persistent Volume Claim
   d) Storage Classes
7) Configurations
   a) ConfigMap
   b) Secrets

Docker Fundamentals

- The Docker Story
- Introduction to Images
- Creating Images
- System Commands
- Docker Networking Basics
- Docker Compose
- Scaling out with Swarm Mode and Kubernetes

DURATION

2 days

3.1.8 MicroServices with Kubernetes

Code
A-KUB2

Overview

This course is designed to teach you about managing application containers, using Kubernetes.

Mastering highly resilient and scalable infrastructure management is very important, because the modern expectation is that your favorite sites will be up 24/7, and that they will roll out new features frequently and without disruption of the service. Achieving this requires tools that allow you to ensure speed of development, infrastructure stability and ability to scale. Students with backgrounds in Operations or Development who are interested in managing container based infrastructure with Kubernetes are recommended to enroll!

By the end of the course successful learners will be able to:

- Containerize an application by creating Docker config files and build processes to produce all the necessary Docker images
- Configure and launch an auto-scaling, self-healing Kubernetes cluster
- Use Kubernetes to manage deploying, scaling, and updating your applications
- Employ best practices for using containers in general, and specifically Kubernetes, when architecting and developing new microservices

OUTLINE

1. Introduction to Microservices
    - What is a microservice?
    - Microservice architecture characteristics
    - Microservices
    - Microservice with Spring Boot
    - IaaS & Microservices
    - Best practices for microservice applications
    - Spring Boot
    - Decomposition & Microservices
    - Spring Rest Services
    - Microservices & Devops
    - Spring Boot Actuator
    - Service Discovery
    - Client Resiliency patterns
    - Service Routing
2. Introduction to Docker
    - Docker Introduction
    - Docker Usage
    - Container Volumes & Layers
    - Docker Network management
    - Dockerfile
    - Manage Containers
    - Packaging Software for Distribution
    - Docker Compose
    - Build Automation
3. Kubernetes Fundamentals
    - Introduction: What is Kubernetes,What problems solved, Kubernetes vs Swarm

- Architecture: Kubernetes Architecture, Master - Node Communication
- Key Concepts: Namespaces, Pods, Services, Labels and Selectors
- Workloads: Deployment, StatefulSet, DaemonSet, Job, CronJob
- Services: Services, Ingress, Ingress Controllers
- Storage: Volumes, Persistent Volumes, Persistent Volume Claim, Storage Classes
- Configurations: ConfigMap, Secrets

DURATION

3 days

3.1.9 Kubernetes for Enterprise Operations

Code
A-KUB3

Overview
Kubernetes is quickly becoming the de-facto standard to operate containerized applications at scale in the data-center. This course covers the fundamentals needed to understand Kubernetes and get quickly up-to-speed, to start building distributed applications that will scale, be fault-tolerant and simple to manage. From understanding its origin, to its high-level architecture, powerful API and key primitives, this course takes you from nothing to being in a position to start building complex applications.

1. Networking & CNI
    a. Overview
    b. Kubernetes Concept
    c. The Kubernetes Networking Model
    d. Benchmark
    e. Adding Calico to Kubernetes
2. Scheduling
    a. Overview
    b. Scheduler Configuration
    c. Affinity/Anti-Affinity Rules
    d. Taints and Tolerations
    e. Disruptions
    f. Specifying a PodDisruptionBudget
3. API Objects
    a. API Objects
    b. The v1 Group
    c. API Resources
4. Limiting Resources & Auto-Scaling
    a. Overview
    b. Resource Requests and Limits
    c. Resource Quotas

    d. Horizontal Scaling
    e. Vertical Scaling
    f. Cluster Scaling
    g. Custom-metrics Scaling
5. Helm
    a. Overview
    b. Helm
    c. Using Helm
6. Installation & Configuration
    a. Getting Started With Kubernetes
    b. Kubeadm
    c. Providers
7. Monitoring,Logging and Troubleshooting
    a. Overview
    b. Troubleshooting and Debugging
    c. Logging
    d. Monitoring
8. Security
    a. Overview
    b. Attack Surface
    c. Accessing the API
        i. Transport Level Security
        ii. Authentication
        iii. Authorization
        iv. Admission Controller
    d. Audit Logging
    e. Security Context for a Pod
    f. Pod Security Policies
    g. Configuring Service Accounts For Pods
    h. Network Policies

DURATION

2 days

3.2 Web Technologies

3.2.1 Developing Client-Site Web Applications with ReactJS

Code

D-RCT

Overview

React is a library developed by Facebook, and it is designed to build large applications with data that changes over time. Developers can use ReactJS to create user interfaces (UI) with high performance, where React would automatically manage all UI updates. React is all about building reusable web components, and it also renders on the server using Node.js.

Course Content

- Quick Start
- EcmaScript6
- Npm
- Webpack
- Babel
- ReactJS
- Styling
- ReactRouter
- Redux
- Redux Saga
- Mock JSon Server
- Testing React
- React Intl
- Component Library

Audience

This course is in advanced series and designed for web developers, software & system architects

Duration

3 days

Format

Instructor Lead

Prerequisites

A fundamental knowledge of JavaScript, Ajax, HTML and CSS are required

3.2.2 Next-generation Web Applications with full stack JavaScript

Code

D-N-WEB

Overview

The Modern Web Development course teaches how to develop modern web applications using a combination of JavaScript frameworks. You will learn how to apply these various techniques to create modern client-side web applications that use the power of the client via the rich document object model and the JavaScript runtime environment the browser provides.

Description

1. Client Side
    a. ReactJS
        - Quick Start
        - EcmaScript6
        - Npm
        - Webpack
        - Babel
        - ReactJS
        - Styling
        - ReactRouter
        - Redux
        - Redux Saga
        - Mock JSon Server
        - Testing React
        - React Intl
        - Component Library
2. Server-Side
    a. NodeJS

b. Restful APIs
c. JSON, JSON Mapping

3. NoSQL Database: MongoDB

Audience

This course is in advanced series and designed for web developers, software & system architects

Duration

3 days

Format

Instructor Lead

Prerequisites

A fundamental knowledge of JavaScript, Ajax, HTML and CSS is required.

3.3 Spring Technologies

3.3.1 Developing Enterprise Applications using Spring Framework and JPA

Code

D-SPRING-381-001

Overview

This advanced course is designed for Java developers who need to learn how to use the Spring Framework to create well-designed, testable business applications in an agile manner. During this course, you will learn Spring's features and how to use them. You will also become familiar with the fundamental architectural issues you will need to be aware of when developing with the Spring framework. It is important to know how to use certain parts of the Spring framework, but even more important to be able to decide when to use them!

Topics Covered:

- Work with the Spring Inversion of Control (IoC) Container
- Effectively use JDBC , Hibernate and JPA for data access
- Use JUnit, Spring, stubs and mocking frameworks to effectively implement automated unit and integration tests
- Take advantage of Aspect-Oriented Programming (AOP) to keep code clean and maintainable
- And much more...

Description

Introduction

- Spring Projects
- Basic environment

Spring Architecture

- Spring Framework definition
- Spring Framework design principals

Spring setup

- Setting classpath and jar files
- Setting configuration

Spring Core Concepts

- Spring quick start
- Writing bean definitions
- Configuring objects
- Creating an application context

Design Patterns

- Inversion of Control
- Dependency Injection

Bean Life-Cycle

- Application context life-cycle
- Initialization of beans
- Using beans
- Destruction
- Other considerations

Testing Spring Application

- Unit Testing vs. Integration Testing
- Unit Testing
- Stubs vs. Mocks
- Integration Testing

AOP with Spring

- Spring AOP

- Implementing Aspect
- Configure the Aspect as a Bean
- Property changes with context
- Useful spring JoinPoint Context
- Context Selecting pointcuts
- Advices with spring Aop
- Advice best practices

Spring Data Access

- Spring in enterprise data access
- Spring resource management
- Spring data access support
- Data access in a layered architecture
- Common data access configurations

Spring JDBC Access

- Spring JDBC Template
- JDBC Template Internals
- Creating JDBC DAO
- Configuring the Application

Spring Transaction Management

- Why use transactions?
- Local transaction management
- Programmatic JTA
- Declarative transactions
- Spring transaction management
- Transaction propagation

Object-Relational Mapping with Spring and JPA

- Object-Relational Mapping - ORM
- Java Persistence API – JPA
- Configuration and Project Setup
- Relational Mapping
- Query and JPQL
- Spring with JPA

- Criteria API

Introducing Spring Data Project

- Persistence with Spring Data JPA
- Spring Data Project
- Working with Repositories
- Defining Query Methods
- Query Creation from Method Names
- Custom Repositories

Spring MVC

- MVC and Spring MVC
- Spring MVC configuration
- Dispatcher Servlet, request life cycle
- Creating controllers
- Mapping requests to controllers
- Validating input

RESTful Services

- RESTful web services with Spring MVC

Spring Boot

- Spring Boot Overview
- Installations
- Configuration
- Starters
- A Simple example. Running example
- Creating executable jar

Audience

Developers who aim to develop Java applications within the Spring framework.

Duration

3 days

Format

Instructor Lead

Prerequisites

Core Java Syntax - (D-EJAVA-301-001) Effective JAVA Programming..

3.3.2 Spring Boot

Code

D-SPR-BOOT

Overview

This course will introduce you to Spring Boot which takes an opinionated view of building Spring applications and gets you up and running as quickly as possible.

Topics Covered:

I. Introduction to Spring Boot
   a. Introducing Spring Boot
   b. Installing Spring Boot
   c. Developing your first Spring Boot application
II. Using Spring Boot
   a. Structuring your code
   b. Configuration Classes
   c. Running Application
III. Customizing Configuration
IV. Testing with Spring Boot
V. Spring Boot Actuator: production-ready features
VI. Microservices with SpringBoot
VII. Deploying Spring Boot applications

Audience

This course is in advanced series and designed for senior developers, software & system architects, project managers involved with the development of Spring Boot applications

Duration

3 days

Format

Instructor Lead

Prerequisites

Prior experience of Spring Framework and Maven are required

3.4 Java Technologies

3.4.1 Developing Object-Oriented Programs in Java

Code

D-JAVA-201-001

Overview

This intermediate course uses an example-based approach to provide an overview of the object-oriented paradigm and to illustrate the evolutionary development approach supported by Java™. At the end of this course you will be familiar with the core components and packages of the Java™ Standard Edition and you will be able to apply object-oriented programming principles with Java™, Java™ syntax and semantics. You will have a clear understanding of advanced Java™ topics and Java new features.

Course Content

1. Setup Development Environment
    1.1. Installing Java Standart Edition (JDK)
    1.2. Introduction to Eclipse
    1.3. Installing and running eclipse
    1.4. Using Eclipse as development environment
2. Object Oriented Concepts
    2.1. Encapsulation, Inheritance and Polymorphism

2.2. OO analysis and design: "Is a" and "Has a"

2.3. Designing an OO application step by step

3. Java SE Language Fundamentals
4. Primitive Data Types
5. Control Statements
6. Classes and Methods
7. Type Casting
8. Inheritance
9. Interfaces
10. Core Class Library
11. Collections and Streams
12. Exception Handling
13. Generics,Compile-time type safety.
14. Enhanced Iterators
15. Autoboxing/Unboxing
16. Typesafe Enums
17. Varargs
18. Annotations
19. JDBC
20. JavaDoc
21. Junit
22. Debugging

Audience

This course is designed for developers, software and system architects and project managers involved with the development of Java™ applications.

Duration

4.5 days

Format

Instructor Lead

Prerequisites

Experience in the following areas is required: Some prior programming experience in a procedural or object-oriented language.

3.4.2 Effective JAVA Programming Using Design Patterns

Code

D-EJAVA-301-001

Overview

This is an advanced Java™ Programming training course that teaches Java developers how to use design patterns and the latest advanced Java language skills effectively.

With the advent of Java 5 and Java 6, the language has seen profound improvements of which not all developers are aware. This course highlights those improvements, as well as delving into a range of topics that an experienced Java developer needs, such as design patterns, performance (JVM, Garbage collection,memory leak, profiling etc), concurrency and refactoring issues: skills that underpin best Java development project practice worldwide.

Topics include:

- Study of best Java development project practice worldwide
- Effective use of advanced Java language constructs
- Study and applications of over 20 Design Patterns
- Performance, concurrency and refactoring

Description

Applying OO Concepts with Java

- Objects and Messages
- Encapsulation, minimizing accessibility and mutability
- Polymorphism
- Subtyping and Subclassing
- Composition versus Inheritance
- Design patterns with Java - an overview
- Structural patterns
    - The Composition pattern
    - The Adapter pattern
    - The Bridge pattern

- The Decorator Pattern

Effective Java

- Creating and destroying Objects
- Creational Design Patterns
  - The Factory Method provides
  - The Abstract Factory Method
  - The Builder Pattern
  - The Prototype Pattern
  - The Singleton Pattern

- Common Methods and why they are important
  - equals and hashCode
  - toString
  - cloning, deep and shallow copying
  - comparing object

- Managing object behavior with patterns
  - Events and changes
  - The chain of responsibility
  - The Observer pattern
  - The Mediator defines
  - The Chain of Responsibility
  - The Command pattern

- Managing Object State and Function
  - The Visitor pattern adds function to a class
  - The State pattern

- Generics
  - Implementing typesafe heterogeneous containers
  - Generic types and methods
  - Lists versus arrays, foreach versus other loops
  - The Iterator pattern

- Enums and Annotations
  - Annotations versus Naming patterns
  - Defining your own annotations

- Exceptions
  - Exceptions for exceptional conditions
  - Programming errors
  - Recoverable conditions
  - Unnecessary usages
  - Standard versus Custom exceptions
  - Documenting Exceptions
  - Capturing failure information
  - Ignored exceptions

- Concurrency
  - Accessing shared mutable data
  - Effective use of synchronization
  - Executors, tasks and threads
  - Lazy initialization and concurrency

Refactoring Java Code

- The basics of refactoring: Detect, characterize, design, modify
- When to refactor
- Tools
- Moving a class
- Extracting methods
- Extracting supertypes
- Conditionals vs Polymorphism

Audience

This course is designed for developers, software and system architects and project managers involved with the development of Java™ applications.

Duration

2 days

Format

Instructor Lead

Prerequisites

A fundamental knowledge of Java is a prerequisite for this course.

3.4.3 Java New Features

Code

D-JAVAN

Overview

With this course, where you will learn the important new Java features introduced in Java 8, Java 9, Java 10 and Java 11. Oracle has announced that after releasing Java 9, they are changing the release cycle scheme to a much faster one. Instead of releasing only major increments twice a decade, we'll be getting Java updates twice a year and a LTS version every three years.

Description

Topics Include:

1. Java8 New Features
   - Lambda Expressions
   - Method enhancements
   - New Functional Interfaces
   - Streams
   - Enhanced Collections API
2. Java9 New Features
   - Java 9 Module System
   - Linking
   - Java 9 REPL (JShell)
   - Collection Factory Methods
   - Private methods in Interfaces
   - Reactive Streams
   - Stream API Improvements
   - New HTTP Client
   - Multi-release JARs
3. Java10 New Features
   - Local Variable Type Inference
   - Consolidate the JDK Forest into a Single Repository
   - Unmodifiable Collections
   - Optional*.orElseThrow()
   - Deprecations and Removals
   - Time-Based Release Versioning
4. Java11 New Features
   - Running Java File with a single command
   - Java String Methods
   - Local-Variable Syntax for Lambda Parameters
   - Nested Based Access Control
   - Dynamic Class-File Constants

- Remove the Java EE and CORBA Modules
- HTTP Client
- Reading/Writing Strings to and from the Files

Audience

Any Java developer who needs to get up to speed with the latest features of Java platform / language

Duration

2 days

Format

Instructor Lead.

Prerequisites

Delegates should be comfortable with Java language, syntax and object oriented application development principles.

3.4.4 Effective Agile Java Development

Code

D-E504

Overview

This is an advanced and pragmatic workshop which will cover the latest agile development practices and tools that are used in Java™ Development.

Description

This course is an advanced pragmatic workshop that teaches latest agile development practices and tools. It provides practical experience across the full scope of agile development activities, including requirements gathering, acceptance test driven development (ATDD), behavior driven development (BDD), test driven development (TDD), agile architecture and design, clean coding practices, continuous integration and agile development teamwork and collaboration. Students will build a small application from the ground up using ATDD and TDD practices and getting exposure to innovative tools such as Maven, Jenkins/Hudson, Subversion, JUnit, Mock Testing, Selenium, Spock, JBehave.

Automated testing techniques are covered in detail in this workshop. Indeed, learning how to write more effective tests is an excellent way to write better designed, more maintainable and more reliable code. The course covers fundamental TDD and BDD practices for Java Developers. Continuous Integration, or CI, is a cornerstone of modern software development best practices.

Topics Include:

- **Apache Maven**

  Apache Maven is a software project management and comprehension tool. Based on the concept of a project object model (POM), Maven can manage a project's build, reporting and documentation from a central piece of information.

  Maven's primary goal is to allow a developer to comprehend the complete state of a development effort in the shortest period of time. In order to attain this goal there are several areas of concern that Maven attempts to deal with:

  - Making the build process easy
  - Providing a uniform build system
  - Providing quality project information
  - Providing guidelines for best practices development
  - Allowing transparent migration to new features

- **Principles of Behavior Driven Development (BDD) and Agile Requirements:**

  An introduction to the fundamental concepts and motivations behind Behavior-Driven Development and related techniques. BDD principles can be applied to requirements gathering and analysis activities. In this module, we learn how techniques such as Acceptance Test Driven Development, Executable Specifications and Feature Injection can significantly improve the quality, reliability and relevance of the application being built, and provide a much better traceability back to, and understanding of, the core business requirements. The course uses high-level BDD tools such as JBehave to build a working set of executable specifications from the ground up.

- **Agile Development Principles**

  Architecture is just as important in Agile projects as it is in conventional software projects. In this module we discuss the key principles of Agile development and design, including the role of architecture in Agile projects, when, how and by whom architecture is specified, implemented and validated.

- **Test Driven Development**

Test Driven Development (TDD) is a key Agile development practice, and is arguably the single most effective way to improve code quality and reliability. In this section, we focus on core TDD and BDD practices at the coding level, and learn how to effectively apply BDD practices in Java both by using advanced JUnit features and testing approaches such as Mock testing. Unit testing vs. Integration testing is covered. Web testing frameworks such as Selenium is introduced. Specific BDD libraries such as Spock are covered as well as topics such as good test design and organization and testing databases.

- **Refactoring and Clean Coding**

  Refactoring is an essential part of maintaining high code quality and reducing maintenance costs. And a solid understanding of clean coding principles is vital to writing maintainable and understandable code. This course explores refactoring principles and patterns, and also discusses effective techniques to help make your code clean, readable and highly maintainable.

- **Team Programming and Collaboration**

  Team collaboration and communication techniques are discussed along with source code control and version management systems. Subversion and distributed systems such as Git are covered.

- **Continuous Integration and Delivery**

  This module covers how Continuous Integration and Delivery practices can be used to enhance team communication and accelerate feedback. Jenkins, an Open Source Continuous Integration tool, is by far the most popular . This course will teach you how to build a powerful and robust CI infrastructure using Maven and Jenkins and automating the build process with Jenkins and provide a wealth of best practices and real-world tips

Audience

This course is in advanced series and designed for senior developers, software & system architects, project managers involved with the development of Java™ applications.

Duration

2 days

Format

Instructor Lead.

Prerequisites

A fundamental knowledge of Java is a prerequisite for this course.

3.4.5 Building Secure Applications with Java and JavaEE

Code

D-Secure-java

Overview

This advanced cource is designed for building secure applications with Java and Java EE. Main topics are

- Software Development Life Cycle
- Java Application Security
- OWASP Java Best Practices

Description

Course include:

1. OWASP
2. Web Application Security Consortium
3. OpenSAMM
4. Enterprise Security Concepts
    a. Basic Vulnerability Terminology
    b. Enterprise Security APIs
    c. Software Development Life Cycle & Security Guideline
    d. Software Assurance Maturity Model
5. Security in Software Development Lifecyle
    a. Security Requirements
    b. Threat Modeling
    c. Secure Design Guidelines
    d. Secure Coding Guidelines
    e. Testing for web application security
    f. Secure administration and Security within Change Management
    g. Deployment WebApp Security Controls
    h. Secure Development Life Cycle
    i. Web Application Security Roles and Responsibilities

6. OWASP Top 10 Web Application Security & Vulnerabilities
    a. A1: Injection
    b. A2: Broken Authentication and Session Management
    c. A3: Cross Site Scripting
    d. A4: Insecure Direct Object Reference
    e. A5: Security Misconfiguration
    f. A6: Sensitive Data Exposure
    g. A7: Missing Function Level Access Control
    h. A8: Cross Site Request Forgery (CSRF)
    i. A9: Using Known Vulnerable Components
    j. A10: Unvalidated Redirects and Forwards
    k. A9: Using Known Vulnerable Components
    l. A10: Unvalidated Redirects and Forwards
    m. A9: Using Known Vulnerable Components
    n. A10: Unvalidated Redirects and Forwards
7. Testing for Vulnerabilities
    a. Web Application Security
    b. Software Security Assurance (SSA)
    c. Find Vulnerabilities
    d. Testing for application vulnerabilities
    e. Black Box vs. Gray Box
    f. Tools of the trade
    g. WebGoat
    h. The Zed Attack Proxy
    i. LAPSE+
8. Secure Development Practices
    a. Validating User Input
    b. Authentication
    c. Authorization
    d. Session Management
    e. Using Interpreters
    f. Crypto
    g. Catching Errors
    h. File System
    i. Configuration

j. Web 2.0

9. Java Security Overview
    a. Information Security Principles
    b. Controls for Information Security
    c. Java EE Security Needs
    d. Java EE Security Components
    e. Securing EJBs and Web Applications
10. Enterprise Security API (ESAPI)
    a. ESAPI - Goals
    b. ESAPI to OWASP Top 10 Mapping
    c. ESAPI Maturity
    d. ESAPI Approach
11. SQL Injection Protection
    a. SQL Injection Attacks
    b. Finding SQL Injection Bugs
    c. Mitigating SQL Injection
    d. Methods to prevent SQL Injection

Audience

Java Developers

Duration

2 day

Format

Instructor Lead

Prerequisites

An advanced knowledge of Java is a prerequisite for this course.

3.4.6 Effective Testing for Java developers

Code

D-E505

Overview

The Test-Driven Development training course delivers a hands-on view into how TDD can be used by developers, project managers, and the quality assurance teams to create higher quality software.Ensuring every student has the same understanding of standard software testing procedures. The TDD training course begins with an examination of common test terminologies, practices, benefits and pitfalls. The course then moves into a discussion on the theory and practice of Test-driven development in Java, the applicability of TDD in modern software development paradigms, and how it can be leveraged within different software engineering processes.Students Will Learn:

- Unit Testing Using JUnit
- Test Driven Development (TDD)
- Behavior Driven Development (BDD)
- Mocks and Stubs
- Automated Testing
- Continuous Integration Servers
- TDD Patterns

Course include:

1. Software Testing
    - Software Testing Levels
    - Software Testing Methods
    - Software Testing Types
2. Testable Code
    - What is testable code?
    - Design requirements for testable code
    - Code smells for testable design
    - Top Things Make Code Hard to Test
    - Code samples
3. Test Driven Development (TDD)
    - Introduction to TDD
    - Test First vs Test Last
    - Red, Green, Refactor
    - Crucial Design Principles
    - Driving Design using TDD
4. Unit Testing and JUnit
    - What is Unit Testing?
    - What is JUnit?
    - Test Cases, Test Suites
    - JUnit Annotations
    - Assertions
    - Conditional Test Executions
    - Repeated Tests

- Parametrized Tests

5. TDD Patterns
   - Red Bar Patterns
   - Testing Patterns
   - Green Bar Patterns
6. Mocks, Stubs
   - Mock Objects with Mockito
   - Stubs
7. Test Coverage with JaCoCo
8. Web Interface (UI) Testing
   - Testing Web Interfaces
   - SeleniumRC, Selenium IDE, Selenium WebDriver
   - Selenium Commands
   - Selenium WebDriver API
   - Testing on Different Browsers
   - Page Object Design Pattern
9. Principles of Behavior Driven Development (BDD)
   - Behavior-Driven Development and related techniques
   - TDD vs BDD
   - User Stories
   - Use Cucumber to script and run acceptance tests
10. Continuous Integration/Automated Testing
    - Continuous Integration
    - Continuous Delivery
    - Checking into Repository
    - Continuous Integration Servers (Jenkins)
    - Automate the Build/Deployment
11. Database Unit Testing
    - Writing Database Tests
    - DbUnit
12. Rest API Testing with Java
13. Performance Testing with JMeter

Audience

This course is in advanced series and designed for senior developers, Q/A teams, software & system architects, project managers involved with the development of Java™ applications.

Duration

2 days

Format

Instructor Lead

Prerequisites

A fundamental knowledge of Java is a prerequisite for this course.

3.4.7 Writing Elegant and Readable Code

Code

D-CLN-C

Overview

If you're a developer, then there probably have been times when you've written code and, after a few days, weeks, or months, you looked back at it and said to yourself "What does this piece of code do?" The answer to that question might have been "I really don't know!" In that case, the only thing you can do is going through the code from start to finish, trying to understand what you were thinking when you wrote it

In this course, we will give you practical techniques for writing clean code. We won't be talking about specific architectures, languages, or platforms. The focus lies on writing better code.

Topics Covered:

- CODE SHOULD BE EASY TO UNDERSTAND
    - What Makes Code “Better”?
    - The Fundamental Theorem of Readability
    - Is Smaller Always Better?
    - Does Time-Till-Understanding Conflict with Other Goals
- SURFACE-LEVEL IMPROVEMENTS
    - Packing Information Into Names

- Names That Can’t Be Misconstrued
- Aesthetics
- Knowing What To Comment
- Making Comments Precise And Compact

- SIMPLIFYING LOOPS AND LOGIC
  - Making Control Flow Easy To Read
  - Breaking Down Giant Expressions
  - Variables And Readability
- REFACTORING YOUR CODE
  - Principles in Refactoring
  - Bad Smells in Code
  - Building Tests
  - Toward a Catalog of Refactorings
  - Composing Methods
  - Moving Features Between Objects
  - Organizing Data
  - Simplifying Conditional Expressions
  - Making Method Calls Simpler
  - Dealing with Generalization
  - Extracting Unrelated Subproblems
  - One Task at a Time
  - Writing Less Code
  - Big Refactorings

Audience

Developers who aim to write better code.

Duration

1 day

Format

Instructor Lead

Prerequisites

Any programming language experience.

3.5 Mobile Technologies

3.5.1 Developing Mobile Applications with ReactNative

Code

D-MBRC-001

Overview

This course focuses on developing truly cross-platform, native iOS and Android apps using React Native. React Native uses modern JavaScript to get truly native UI and performance while sharing skills and code with the web. You will learn about UI development with React Native UI and layout support and access the native mobile platform's capabilities from Javascript.

- Getting Started with React Native
- React Native Tools
- Javascript ES6 Overview
- Create your first React Native app
- Developing your UI with JSX
- Going deeper with React Native

Description

1. Intro to React Native
    - What is
    - Why you should use it
    - Abstracting React from the DOM
    - Advantages of React Native

- React Native vs Web Apps
- React Native vs React web

2. Getting Started with React Native
   - Installing React Native
   - iOS setup – XCode
   - Android setup – Android Studio
   - Create an example app
   - Exploring Project Structure
   - Run an example project in iOS and Android simulators
   - Debugging
3. React Native Tools
   - Console + editor
4. Javascript ES6 Overview
5. Core React Native Components
   - Establishing a layout with View
   - Displaying text with Text
   - Introduction to TextInputs
   - Adding images with Image
   - Interactive Design
   - Overview of form validation
   - Submitting & clearing a form
   - Creating custom Components
   - Properties (props)
   - Managing State
   - Populating and Manipulating Lists
   - Using open source (NPM)
   - Making components interactive with TouchableHighlight
   - Displaying data with ListView
   - Changing screens with Navigator
   - Expanding touch capability with GestureResponder and PanResponder
   - Higher-Order components
6. Direct Manipulation
7. Styling
   - Describe StyleSheet, what are available values, how it works
   - Flexbox and its properties
   - Create Immutable style objects with Stylesheet.create
   - External Stylesheets
   - Pass styles as props
   - Positioning components with flexbox
8. ReactNative APIs
   - Using fetch to retrieve data
   - Getting a user's location and handling permissions
   - Accessing stored photos with CameraRoll
   - Adding animations
9. Navigation
   - Basic concepts of navigation.
   - React Native Navigation

10. Working with HTTP, network requests, and accessing restful services
11. Persistence
    - persistence using AsyncStorage
    - available APIs, like `multiSet` and when it's better to use what
12. Animations
    - Introduce `Animated` as a general solution for animations
    - Demonstrate how Animated works
    - Encourage them to animate a few things on the screen
    - Perf. wise - mention native driver
13. Lists
    - How <FlatList /> and <SectionList /> work and when to use each.
    - Available scroll solutions
    - Performance optimizations
14. Redux
15. Redux Sage
16. Using Native UI Components
17. Native Modules
18. Using 3.party libraries
19. Cross Platform APIs
20. iOS specific APIs
21. Android specific APIs
22. Deployment
    - Deploying to Apple App Store
    - Deploying to Android Play Store

Audience

Software and web developers who want to rapidly create and deploy engaging mobile apps that will look great and perform well on a wide variety of devices.

Duration

3 days

Format

Instructor Lead

Prerequisites

Delegates should be comfortable coding JavaScript from scratch, and web fundamentals (HTML and CSS).

3.6 Executive Training

3.6.1 Technology Trends for Executives

Code

E-TRDS

Overview

What skills do new leaders need to have to be successful, and how can they learn them?

The world changes quickly, and as we've seen countless times, any company unable (or unwilling) to go with the flow, can often end up left behind. This is why new leaders are poised to take on the waves of technology, and to do so successfully requires specific skills that need cultivation. Skills that, in the past, may have been considered "nice-to-have," but are now regarded as crucial. So what are these new emerging trends in this modern age?

Description

- Cloud Solutions
- IoT
- Artificial Intelligence
- Blockchain
- Voice-driven Software
- 5G
- 3D Printing
- Advanced Robotics
- Autonomous Vehicles

Audience

non-IT company executives and business owners from all sectors

Duration

1 day

Format

Instructor Lead

Prerequisites

3.7 Analysis and Design

3.7.1 Object Oriented Analysis and Design using UML

Code

M-OOAD-501-001

Overview

This 3-day intensive course is a language neutral introduction of object oriented software development. Object oriented analysis and design techniques will be trained using methods from the Unified Modeling Language (UML). What you will learn At the end of this course you will be familiar with the concepts of object oriented software development, and you will be able to apply object oriented analysis and design methods for simple applications. You will know the most important diagram techniques and will be able to join a development team and actively participate in the design of an application.

Description

- The OO life cycle
- Object oriented concepts in depth
- Encapsulation, Interfaces Inheritance, Polymorphism
- Requirements analysis
- Determine object candidates
- Assigning responsibilities
- CRC-Cards
- Design techniques and diagrams
- Use case diagrams
- Class diagrams
- Component diagrams
- Interaction diagrams
- Activity diagrams
- State/Transition diagrams

- Software design patterns overview
- Frameworks overview
- Software development processes

Audience

Designers, architects, developers and project managers who develop software in an object oriented language like Java or C++ Designers who want to apply proper analysis and design techniques in order to code more robust and better maintainable code *Lectures and demos combined with hands-on exercises using computer-based labs.

Duration

3 days

Format

Instructor Lead

Prerequisites

Experience in application development and basic OO skills such as those from the course eJava111 Developing Object-Oriented Programs in Java™.

3.8 Project Management

3.8.1 Agile Methodology & Scrum

Code

P-AG-001

Overview

Agile Methodologies

- History of Agile
- Agile Methodologies
  - Scrum
  - Kanban
  - Extreme Programming

- Manifesto for Agile Software Development
- Scrum
    - How it works?
    - What are the roles?
        - Team
        - Product Owner
        - Scrum Master
    - What are scrum ceremonies?
        - Release Planning
        - Sprint Planning
        - Sprint Review
        - Sprint Retrospective
        - Daily Scrum
    - Artifacts
        - Product Backlog
            - Creating
            - Managing
            - Sprint Backlog
            - Burn Down Chart
            - Impediment List
- Scrum from the view of Scrum Master
- Scrum from the view of Product Owner
- Scrum from the view of Team
- Estimating & Planning
- Measuring & Reporting

Audience

Duration

2 days

Format

Instructor Lead

Prerequisites

3.8.2 Agile Methodology & Scrum& Agile Testing

Code

P-AG-TST-001

Overview

Agile Methodologies

- History of Agile
- Agile Methodologies
    - Scrum
    - Kanban
    - Extreme Programming
- Manifesto for Agile Software Development
- Scrum
    - How it works?
    - What are the roles?
        - Team
        - Product Owner
        - Scrum Master
    - What are scrum ceremonies?
        - Release Planning
        - Sprint Planning

        - Sprint Review
        - Sprint Retrospective
        - Daily Scrum
    - Artifacts
        - Product Backlog
            - Creating
            - Managing
            - Sprint Backlog
            - Burn Down Chart
            - Impediment List
- Scrum from the view of Scrum Master
- Scrum from the view of Product Owner
- Scrum from the view of Team
- Estimating & Planning
- Measuring & Reporting

Test Methodologies

The Role & Characteristics of a Tester

Test Methods

- Static - Dynamic Methods
- Black Box - White Box
- Visual Testing

Testing Levels

- Unit testing
- Integration testing
- System testing
- Acceptance testing

Test Objectives

- Compatibility Tests
- Smoke Tests
- Regression Tests
- Alpha Tests
- Beta Tests
- Functional Tests
- Non Functional Tests
- Performance Tests
- Usability Tests
- Security Tests

Agile Methodologies & Testing

- What is your "Done" Definition?
- Testing at Agile
- Difference of Testing at non Agile Team & Agile Team
- Test Effort Estimation
- Different Perspectives for Agile Testers
  - Continuous Integration
  - Version Management
  - Pairing
  - Acceptance Criteria
  - Regression Testing
  - Defect Management
  - Test Driven Development
  - Test Automation
  - Agile Performance Testing
- Behaviors that might cause Agile Testing To Fail

- Improvement Process

Audience

Quality & Test Specialists, Developers, Project Managers, Product Owners

Duration

3 days

Format

Intructor Lead

Prerequisites

3.9 Software Project Management

3.9.1 Apache Maven

Code

D-MVN

Overview

This course covers all of the basic fundamentals of Maven. It covers dependencies, plugins, repositories, IDE integrations, and all the basic commands of Maven.

Description

Apache Maven is a software project management and comprehension tool. Based on the concept of a project object model (POM), Maven can manage a project's build, reporting and documentation from a central piece of information.
Maven's primary goal is to allow a developer to comprehend the complete state of a development effort in the shortest period of time. In order to attain this goal there are several areas of concern that Maven attempts to deal with:

- Making the build process easy
- Providing a uniform build system
- Providing quality project information
- Providing guidelines for best practices development
- Allowing transparent migration to new features

Topics Include:

1. Introduction to Maven
    a. What is Maven?
    b. How Maven works?
    c. What does Maven do?
    d. Maven plugin architecture
    e. Conceptual Model of a project
    f. İnstalling Maven
    g. Eclipse and Maven
2. Maven Quickstart
    a. Create a simple Maven project
    b. Archetypes
    c. Project structure

    d. Super pom
    e. Building Maven Project
3. Maven Core Concepts
    a. Maven plugins and Goals
    b. Maven Lifecycle
    c. Maven Coordinates
    d. Dependency Management
    e. Maven Repositories
    f. Site Generation and Reports
4. Customizing a Maven Project
    a. Customizing compiler
    b. Customizing project information
    c. Managing dependencies and scopes
    d. Managing classpath resources
    e. Customizing tests
    f. Integration testing
5. Multi-module Projects
    a. Module Layout Strategies
    b. Parent and Submodule Configuration
    c. Web Projects
    d. Building Multi-Module Projects
6. Dependency Management in Depth
    a. Transitive Dependencies
    b. Limiting Dependencies
        i. Dependency Mediation
        ii. Dependency Scopes
        iii. Dependency Management
        iv. Excluded Dependencies
        v. Optional Dependencies
    c. Importing Dependencies
    d. Bill Of Materials (BOM)
7. Release Management
    a. Releasing Software
    b. Maven Release Plugin
    c. Distribution Repositories
    d. Deploy to Nexus Instance
    e. Introduction to Nexus
8. Maven Best Practices

Audience

Ideal for programmers who want to use Maven on their projects. This class is also appropriate for the existing Maven user who is interested in developing a greater understanding of the Maven fundamentals.

Duration

1 day

Format

Instructor Lead.

Prerequisites

prior experience of JAVA and eclipse

3.9.2 Gradle

Code

D-GRDL

Overview

This course is for build masters and developers who are authoring their builds. Participants will learn how to use the Gradle build system to substantially increase their productivity.

Topics Include:

- Introduction
- Groovy for Gradle
- Installing Gradle
  - Environment variables
  - Testing your installation
- Quick Tour of Gradle
  - Creating build scripts
  - Declaring dependencies
  - Accessing repositories
  - Using plugins
  - Configuring the directed acyclic graph
- Building Java projects
  - Standard project structure
  - The Java plugin
  - Running tests
- Building Groovy projects
  - The Groovy project structure
  - Working with both Groovy and Java
  - Executing tests with both JUnit and Spock

- Defining Tasks
  - Declaring tasks
  - Defining project properties
  - Setting dependencies
  - Using doFirst and doLast
  - Using the built-in task types
- The Gradle Daemon
  - Usage and troubleshooting
  - Configuring the daemon
- Web projects
  - Standard web layout
  - The war and jetty plugins
  - Customizing web projects
- IDE Integration
  - Eclipse projects
- The Gradle wrapper
  - Specifying versions
  - Generating the scripts
- Multi-project builds
  - Using settings.xml
  - Consolidating configuration properties
  - Making one subproject depend on another

Audience

Ideal for programmers who want to use Gradle on their projects. This class is also appropriate for the existing Gradle user who is interested in developing a greater understanding of the Gradle fundamentals

Duration

2 days

Format

Instructor Lead.

Prerequisites

prior experience of JAVA and eclipse

3.10 SQL

3.10.1 PL/SQL Programming

Code

D-SQL

Description

Topics in this course :

- What is SQL?
- What is PL/SQL?
- Data Types
- Table basics
- Creating tables
- Selecting data
- Inserting records
- Updating records
- Deleting records
- Drop/alter table
- Primary Keys
- Foreign Keys
- Creating Index
- Some important SQL Functions
- SQL Joins

- SQL Functions, Procedures and Triggers
- Some Important performance issues

Audience

Duration

1 day

Format

Instructor Lead

Three-day instructor-led class with approximately 50% hands-on labs

Prerequisites