مسیر آموزش
برنامه‌نویسی جاوا

طول دوره: ۷۷ ساعت

مدرس: احمدرضا صدیقی

سرفصل‌های مسیر آموزش برنامه‌نویسی جاوا

۱. مقدمه و آشنایی با جاوا:

- تاریخچه جاوا
- چگونگی کارکرد جاوا
- خانواده جاوا
- معرفی JDK، JRE و JVM
- آماده کردن محیط برنامه نویسی جاوا در ویندوز و لینوکس
- نوشتن اولین برنامه جاوا، کامپایل و اجرا
- پنجره دستور و ورودی و خروجی استاندارد
- پیاده سازی چندین متد
- پیاده سازی چندین کلاس
- نصب IntelliJ IDEA

در این جلسه، با چگونگی کارکرد جاوا و تفاوت‌های جاوا با زبان‌های محلی (native) مثل C و C++ آشنا می‌شوید و فرایند «کامپایل» و «اجرای» برنامه‌های جاوا را می‌آموزید. همچنین می‌آموزید که چگونه محیط برنامه نویسی جاوا را در ویندوز و لینوکس آماده کنید و یک برنامه ساده به زبان جاوا بنویسید و اجرا کنید. در انتها نیز با نصب IntelliJ IDEA و استفاده از آن به عنوان ویرایشگر و محیط برنامه نویسی (IDE) جاوا آشنا می‌شوید.

۲. مفاهیم پایه:

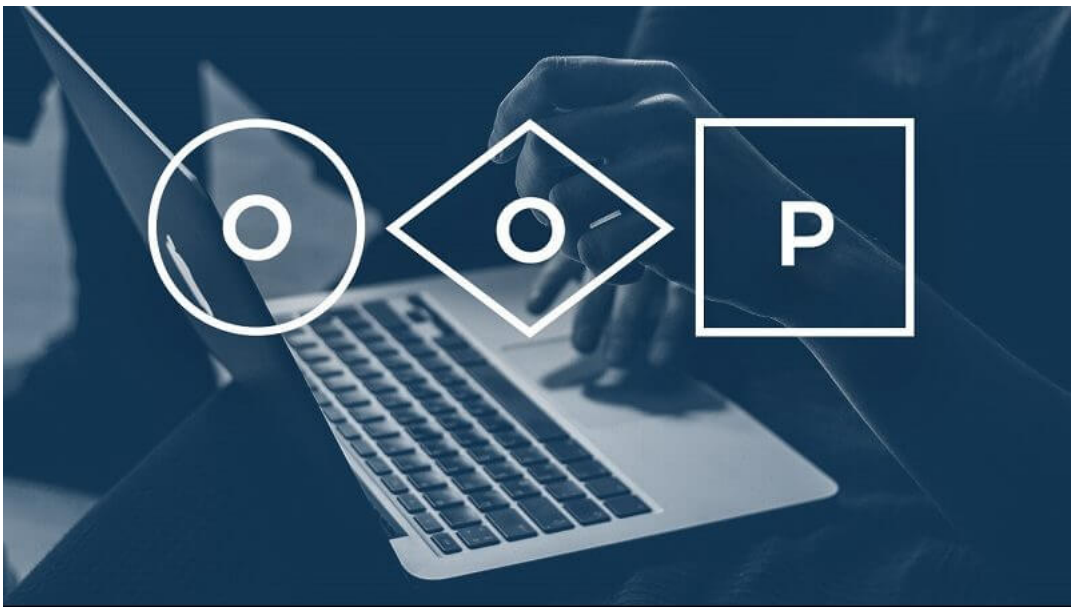
- انواع داده‌ها
- متغیرها و قوانین نامگذاری آن‌ها
- محدوده قابل مشاهده متغیرها
- عملگرها (محاسباتی، مقایسه‌ای، منطقی، بیتی، انتساب)
- ساختارهای کنترلی (if، if-else، for، while، do-while، switch-case)
- کلمات کلیدی return و continue
- آرایه‌ها

در زبان جاوا انواع داده‌ها قابل استفاده اند. داده‌های عددی صحیح، عددی اعشاری، و کاراکتر از جمله این داده‌ها هستند. روی این داده‌ها می‌توانید مجموعه‌ای از عملیات را انجام دهید که انجام این عملیات با استفاده از عملگرهای مختلف این زبان (محاسباتی، مقایسه‌ای، منطقی، بیتی، انتساب) قابل انجام هستند. در جاوا از ساختارهای کنترلی برای کنترل اجرای برنامه، بررسی شرایط، یا تکرار عملیات استفاده می‌شود. در پایان، با آرایه‌های دو بعدی و چندبعدی و نحوه تعریف و مقداردهی آنها در زبان جاوا آشنا می‌شوید.

۳. شی گرایی و ارث‌بری

- شی گرایی و ارث‌بری
- افزودن متد به کلاس
- سازنده
- Finalize ()
- متدهای ToString () و Equals ()\
- کلمه کلیدی this
- Overloading
- تعریف پکیج (package)
- فیلدهای static و final

شی گرایی روش اصلی در توسعه اکثر قریب به اتفاق نرم افزارهای امروزی است. اهمیت شی گرایی به اندازه‌ای است که اگر یک برنامه نویس، شی گرایی را نداند می‌توان گفت او برنامه نویس نیست. متاسفانه بسیاری از برنامه نویسان امروزی درک دقیق و کاملی از شی گرایی ندارند و شی گرایی را به صورت کلی و غیر دقیق بلد هستند. در این جلسه، علاوه بر مفاهیم شی گرایی، کاربرد و موارد استفاده هر یک از مفاهیم و جزییات شی گرایی با مثال‌هایی کاربردی و عملی به شما آموخته می‌شود تا پس از این جلسه درک دقیق و کاملی از شی گرایی به دست می‌آورید.



ارث بری بین کلاس‌ها، چرا و چگونه؟

- پیاده سازی سازنده در کلاس فرزند
- کلاس Abstract
- کلاس Final
- کلاس java.lang.Object

ایده ارث بری از دنیای واقعی گرفته شده است، زیرا موجودات دنیای واقعی، شباهت‌های زیادی به هم دارند به گونه‌ای که می‌توان تصور کرد این خصوصیات را از یکدیگر به ارث برده‌اند. از آنجاییکه هدف شی گرایی، شبیه سازی دنیای واقعی است، ارث بری به شی گرایی نیز راه پیدا کرده است. در این جلسه، با ارث بری و اینکه چگونه به ما در طراحی، تولید و نگهداری برنامه کمک می‌کند آشنا می‌شوید. مثل همیشه، مثال‌های کاربردی و عملیاتی را برای توضیح و پیاده سازی ارث بری ملاحظه خواهید کرد.

۴. اینترفیس و Enumeration

- اینترفیس چیست و چگونه تعریف می‌شود؟
- پیاده سازی اینترفیس
- چند نکته در رابطه با اینترفیس‌ها
- ارث بری میان اینترفیس‌ها
- Enum چیست و چرا؟
- جزییات پیاده سازی Enumها

اینترفیس‌ها یکی دیگر از مفاهیم شی گرایی است که طراحی برنامه‌های جاوا را ساده و انعطاف پذیر می‌کند.
Enum مکانیسمی برای تعریف مقادیر ثابت در برنامه‌های جاوا هستند که در این جلسه با آن آشنا می‌شوید.

۵. کنترل خطا و استثنا

- مفهوم استثنا
- استفاده از Exception در کنترل خطا و استثنا
- توسعه کلاس Exception
- تولید بیش از یک Exception در یک متد
- بلوک Finally
- اینترفیس AutoClosable
- متدهای کلاس Exception
- تولید خطاهای زنجیره‌ای
- متدهای (GetMessage)، (GetCause) و (PrintStackTrace)
- استثناهای Runtime و Non-Runtime

در تولید برنامه‌ها، همواره سناریوهای خوشبینانه مدنظر هستند. اما وقتی یک برنامه در محیط عملیاتی قرار می‌گیرد، شرایط محیطی، وضعیت داده‌ها، نحوه تعامل یک کاربر با برنامه منجر به شرایط و استثناهای مختلفی می‌شوند که ممکن است روال اجرای برنامه را از وضعیت نرمال خارج کنند. در چنین شرایطی برنامه می‌بایست شرایط غیر نرمال را کنترل کند و عکس العمل مناسب نشان دهد. کنترل استثنا و خطا بخش مهمی از برنامه نویسی جاوا و زبان‌های دیگر است و بدون آن، تولید هیچ برنامه‌ای (ساده یا پیچیده) غیرممکن است. طراحی درست و استفاده بجای کنترل خطا و استثنا، توسط بسیاری از برنامه نویسان انجام نمی‌شود و در این جلسه شما با اصول و روش صحیح پیاده سازی آن آشنا خواهید شد.

۶. پکیج java.lang

- کلاس‌های Wrapper (Integer, Double, Long, Character,...)
- کلاس System
- کلاس Object
- کلاس Cloneable
- کلاس Class
- کلاس Math

مهم‌ترین بخش از جاوا را مجموعه‌ای از کلاس‌ها تشکیل می‌دهند که در این جلسه به آن‌ها پرداخته می‌شود.

۷. کاراکترها و رشته‌ها

- کلاس String و سازنده‌های آن
- متصل کردن رشته‌ها
- متدهای کلاس String
- مقایسه کردن Stringها
- مکان‌یابی درون رشته‌ها
- استخراج رشته‌های زیرمجموعه یک رشته
- کلاس String Buffer

کار با رشته‌ها و متن‌ها نیاز هر برنامه است. داده‌هایی که توسط کاربر وارد می‌شوند، داده‌هایی که در پایگاه داده ذخیره و بازیابی می‌شوند، یا داده‌هایی که از یک فایل متنی خوانده می‌شوند یا در یک فایل متنی نوشته می‌شوند، همگی متن و رشته هستند و برنامه باید بتواند آن‌ها را نگهداری و پردازش کند. در این جلسه با نحوه تعامل با رشته‌ها آشنا می‌شوید.

۸. ساختمان داده‌ها و پکیج java.util

- Collectionها
- اینترفیس Collection
- اینترفیس List
- اینترفیس Set
- اینترفیس SortedSet
- اینترفیس Queue
- کلاس‌های Collection
- کلاس Array List
- کلاس Linked List
- کلاس HashSet
- اینترفیس Iterator
- حلقه For
- Map
- اینترفیس Map
- اینترفیس Sorted Map
- اینترفیس Map.Entry
- کلاس های Map
- کلاس HashMap
- کلاس Arrays
- کلاس و اینترفیس‌های قدیمی در پکیج java.util
- اینترفیس Enumeration
- کلاس Vector
- کلاس Stack
- کلاس Hashtable
- کلاس Properties
- کلاس Date
- کلاس Calendar
- کلاس Random

در کار با داده‌ها، موارد فراوانی وجود دارد که نیاز است داده‌ها را جستجو کنیم، یا آنها را مرتب کنیم، یا داده‌های تکرار را حذف کنیم یا به سرعت به آنها دسترسی پیدا کنیم. به این منظور مجموعه‌ای از ساختمان داده‌های آماده در زبان جاوا پیاده سازی و آماده شده است که در این فصل با آنها آشنا می‌شوید. ساختمان داده‌های معروفی از قبیل List (که داده‌های خود را به ترتیب نگهداری می‌کند) یا Set (که داده‌های غیرتکراری را نگهداری می‌کند) از معروف ترین اینها هستند. در این جلسه همچنین با کلاس‌های Date و Calendar که برای کار با تاریخ و زمان استفاده می‌شوند آشنا می‌شوید

۹. Generics و ورودی و خروجی (File/Stream)

- Generics چرا و چگونه؟
- نامگذاری پارامترهای Generics
- متدها و سازنده‌های Generics
- استفاده از نشانه‌های ?، extends و super
- ارث بری

یکی از قابلیت‌هایی که انعطاف پذیری کدهای جاوا و استفاده مجدد (Reusability) آن‌ها را افزایش می‌دهد مفهوم Generics است که در این جلسه با ارایه مثال‌های کاربردی و عملی آشنا می‌شوید.

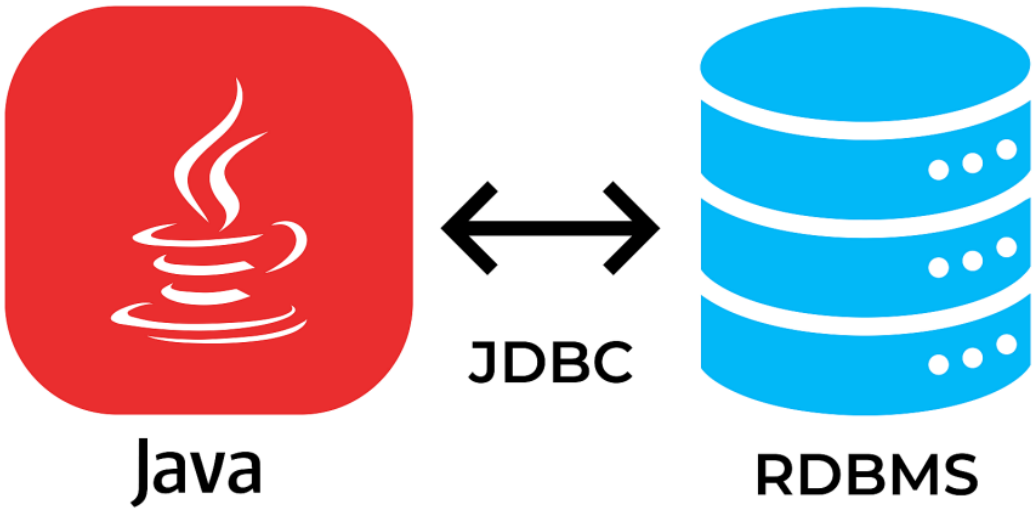
- کار با فایل‌ها و دایرکتوری‌ها با استفاده از کلاس File
- خواندن یا نوشتن در فایل‌ها با استفاده از Streamها
- بافر کردن داده‌ها به منظور کارایی بالاتر
- Serialization
- کلاس Scanner

امروزه تقریبا در تمام برنامه‌ها با فایل‌ها سروکار داریم. ممکن است تنظیمات یک برنامه را در یک فایل ذخیره کنیم، یا بخواهیم فایل‌های عکس، فیلم یا صوت را در برنامه بارگذاری و نمایش دهیم. در این فصل چگونگی کار ایجاد فایل یا دایرکتوری، خواندن فایل‌ها یا نوشتن در آنها را خواهید آموخت. علاوه بر فایل‌ها، خواندن یا نوشتن داده‌ها را در منابع مختلف (مثلا حافظه سیستم) یاد می‌گیرید. در انتها با نحوه سریال کردن داده‌های جاوا برای عبور از شبکه یا ذخیره در فایل را خواهید آموخت.

۱۰. Threadها و همزمانی و ارتباط با پایگاه داده (JDBC)

- موازی سازی بخش‌های برنامه
- کنترل همزمانی با استفاده از Synchronization
- اجرا کننده‌ها
- اینترفیس Callable
- اینترفیس Future
- کلاس Timer و TimerTask

یکی دیگر از مفاهیم مهم در برنامه‌های امروزی، موازی سازی اجرای بخش‌هایی از برنامه است تا به این ترتیب کارایی برنامه و رضایت کاربر افزایش یابد. تصور کنید در زمانیکه کاربر یک فایل را آپلود می‌کند (که زمان زیادی صرف می‌کند)، برنامه می‌تواند به صورت همزمان کار دیگری را نیز انجام دهد. در این فصل استفاده از Threadها و مفاهیم پیاده سازی آنها را در مثال‌های عملی و کاربردی ملاحظه خواهید کرد و چالش‌ها آن‌ها را خواهید آموخت.



- آشنایی با SQL
- مفاهیم JDBC (Driver, Connection, Statement)
- جستجو در پایگاه داده (دستور select و join)
- به‌روزرسانی پایگاه داده (insert، update، delete)
- مدیریت تراکنش
- کلاس Entity

کار با پایگاه داده‌ها، بخش اصلی و همیشگی برنامه‌های امروزی است. بدون پایگاه داده، تقریبا هیچ برنامه‌ای امروزه تولید نمی‌شود! در این جلسه یاد می‌گیرید که چگونه در یک برنامه جاوا به پایگاه داده متصل شوید، داده‌هایی را در آن ذخیره یا بارگذاری کنید. نگران نباشید، برای کار با پایگاه داده نگاهی اجمالی به SQL و مفاهیم جدول، ستون، کلید اصلی، و کلید خارجی خواهیم انداخت، همچنین مفهوم مهم «مدیریت تراکنش» و چگونگی انجام آن در برنامه‌های جاوا را یاد خواهید گرفت.

۱۱. عبارت‌های با قاعده (Regular Expression):

- تعریف قاعده برای رشته‌ها
- کاراکترهای خاص
- محدوده کاراکترها
- میان‌بُرها

- تکرار
- گروه بندی کاراکترها
- ارجاع به عقب
- انطباق در نقاط مرزی
- کلاس Pattern
- کلاس Matcher
- کلاس Pattern Syntax Exception

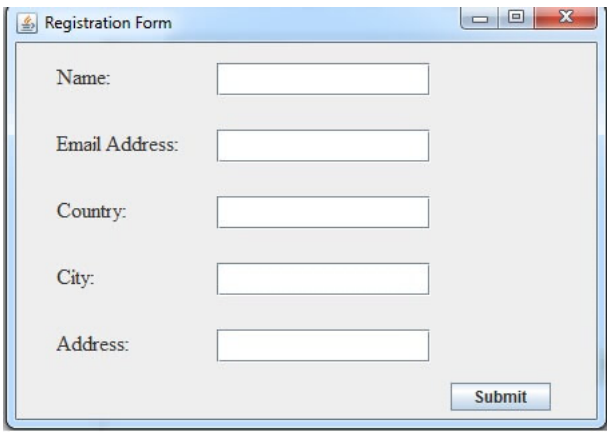
رشته‌هایی که در برنامه‌ها پردازش می‌شوند ممکن است همگی یک قالب داشته باشند، مثلا آدرس وب سایت‌ها، ایمیل‌ها، شماره تلفن‌ها، کد پستی‌ها، همگی یک قالب دارند، به این قالب‌ها «عبارت‌های باقاعده» گفته می‌شود. برنامه‌ها باید بتوانند انطباق یک رشته را بر یک عبارت با قاعده بررسی کنند، یا در یک رشته جستجو کنند و تمام رشته‌های منطق بر یک عبارت را بیابند. در این فصل کار با عبارت‌های باقاعده که اهمیت زیادی در کار با رشته‌ها دارند را خواهید آموخت.

۱۲. Annotation و واسط کاربری (Swing)

- Annotation چرا و چگونه
- انواع Annotation
- عناصر Annotation
- محدوده قابل استفاده از یک Annotation
- سیاست‌های ماندگاری

Annotationها نوعی علامتگذاری در کدهای برنامه هستند، علامت‌هایی که توسط جاوا فهمیده و پردازش می‌شوند. امروزه تمام Java Enterprise مبتنی بر Annotationهاست، بنابراین درک و نحوه پیاده سازی Annotationها اهمیت بسیاری در یادگیری Java Enterprise دارد، البته اگر قصد یادگیری Java Enterprise را داشته باشید!

- کلاس JFrame
- چیدمان نمایش اجزای واسط کاربری (Layout Management)
- کامپوننت‌های واسط کاربری
- منوها



واسط کاربری بخش مهمی از اغلب برنامه‌هاست. تکنولوژی Swing یکی از تکنولوژی‌های جاوا استاندارد برای تولید واسط کاربری است که در این جلسه جزئیات آن‌را خواهید آموخت.
تکنولوژی Swing یک خصوصیت ممتاز به نام «Look And Feel» دارد که امکان می‌دهد تِم یا پوسته نمایش واسط کاربری را بدون تغییر در کدهای برنامه تغییر دهیم. به این ترتیب، وقتی واسط کاربری یک برنامه را توسعه می‌دهید بدون اینکه به شمایل یا زیبایی واسط کاربری فکر کنید، صرفا کارکرد آنرا پیاده سازی و تست می‌کنید، بعدا می‌توانید هر پوسته‌ای بخواهید برای آن انتخاب کنید.یک نکته جالبِ دیگر در مورد واسط کاربری Swing این است، که پنجره‌ها، منوها، دیالوگ‌ها و تمام اجزای واسط کاربری در لینوکس، ویندوز یا مک یکسان و به یک شکل نمایش داده می‌شود.

۱۳. Logging

- Logging چیست؟
- آبجکت Logger
- سطوح لاگ
- آبجکت Handler
- آبجکت Formatter
- آبجکت LogManager

در برنامه های عملیاتی که با داده‌های مهم و ارزشمند کار می‌کنند، ردیابی تمام عملیات برنامه، (ورود و خروج کاربران، ثبت داده، ...) حوادث و رخدادهای برنامه (خطاها و استثناها)، و تغییرات داده (تغییر رمز کاربر، ویرایش پروفایل،...) اهمیت زیادی دارد. Logging یک مکانیسم کارآمد و بسیار مناسب برای ثبت رویدادها و عملیات مختلف برنامه است. با استفاده از Logging، می‌توانید تمام حوادث برنامه را بدون اینکه تاثیر قابل توجهی در کارایی و کارکرد برنامه داشته باشد در فایل‌های متنی که به آنها فایل‌های لاگ گفته می‌شود ذخیره کنید تا بعدا به آنها مراجعه کنید و روند حوادث را در صورت نیاز بررسی کنید.

۱۴. عبارت‌های Lambda و کلاس‌های تو در تو

- کلاس داخلی
- کلاس داخلی استاتیک
- کلاس داخلی غیر-استاتیک
- کلاس محلی
- کار با کلاس بیرونی
- کلاس بی نام (Anonymous)
- Listener
- عبارت‌های لامبدا (Lambda)

- اینترفیس Functional
- کلاس Stream
- متدهای پیش فرض و متدهای استاتیک
- قواعد متدهای پیش فرض
- متدهای استاتیک

مدل سازی بسیاری از موجودات دنیای واقعی در نرم افزار صرفا با استفاده از مفاهیم پایه در شی گرایی دشوار است به همین دلیل جاوا، امکان تعریف یک کلاس داخل یک کلاس دیگر را امکانپذیر کرده است. در این جلسه، کاربرد و اهمیت چنین قابلیتی را با چندین مثال کاربردی بحث خواهیم کرد و سپس استفاده از عبارت‌های لامبدا در «برنامه نویسی رویه‌ای» را خواهید آموخت. در انتها، با متدهای پیش فرض و استاتیک و کاربرد آنها آشنا خواهید شد.

۱۵. Reflection و چند زبانی

- واکاوی جزییات کلاس‌ها
- دسترسی به اجزای داخلی کلاس‌ها (فیلدها، متدها و سازنده‌ها)
- مثال کاربردی

Reflection یکی از تکنولوژی‌های جذاب زبان جاواست که امکان می‌دهد با استفاده از برنامه نویسی، جزییات کلاس‌ها را واکاوی کنیم، آنها را دستکاری کنیم یا در زمان اجرای برنامه آنها را فراخوانی کنیم. قسمت عمده‌ای از Java EE مبتنی بر Reflection است. دانستن Reflection به درک و یادگیری مفاهیم پیشرفته جاوا کمک می‌کند.

- فایل‌های Resource Bundle
- آبجکت‌های Locale
- واکشی متن از آبجکت ResourceBundle
- متن‌ها و پیام‌ها

- فرمت دهی
- استفاده از فرمت‌های آماده
- فرمت‌های دلخواه برای تاریخ و زمان
- تغییر نمادهای تاریخ و زمان
- پیام‌ها

بسیاری از نرم افزارهایی که امروزه تولید می‌شوند به بیش از یک زبان (فارسی، عربی، انگلیسی،...) کار می‌کنند. کاربر می‌تواند در صفحه اصلی برنامه زبان مورد نظر خود را انتخاب کند، و تا آخر با زبانی که انتخاب کرده با برنامه کار کند. متن‌های صفحات، چیدمان صفحات (راست-چین یا چپ-چین)، تقویم صفحات، واحد پول، ...مطابق با زبانی که کاربر انتخاب می‌کند به کاربر نمایش داده خواهد شد. نکته جالب این است که برای تمام زبان‌ها فقط یک کد اجرا می‌شود، این خاصیت در این جلسه آموزش داده می‌شود.

۱۶. ابزار JAR و JavaDoc

- JAR، مکانیسم بسته بندی برنامه‌های جاوا
- فایل Manifest
- اجرایی کردن فایل JAR

وقتی یک برنامه جاوا توسعه داده و تولید می‌شود، مجموعه‌ای از فایلها به وجود می‌آیند. ولی وجود چندین (صدها و در بعضی مواقع هزاران) فایل که در واقع بخش‌های یک برنامه هستند می‌تواند کار نگهداری و حمل و نقل آنها را دشوار کند. ابزار JAR امکان می‌دهد تا فایل‌های یک برنامه را در قالب یک فایل (با پسوند .jar) تجمیع کنیم. فایل JAR مشابه یک فایل ZIP است که خود از مجموعه‌ای از فایل‌ها و دایرکتوری‌ها تشکیل شده است. در این جلسه جزییات ایجاد یک فایل JAR و اجرایی کردن آن را خواهید آموخت.

- Javadoc چیست؟
- علامت‌های قابل استفاده در Javadoc
- تولید مستندات از روی Javadoc

مستندات سازی یکی از بخش‌های با اهمیت نرم افزار است. اما در بسیاری از مواقع وظیفه برنامه نویسان نیست! JavaDoc نوعی از مستندات است که توسط برنامه نویسان لابلای کدها نوشته می‌شود. این مستندات مشابه توضیحات (comment) لابلای کدها نوشته می‌شوند ولی توسط یک ابزار با نام javadoc (یکی از ابزارهای JDK) قابل شناسایی هستند. ابزار javadoc می‌تواند مستندات JavaDoc که داخل کدهای برنامه قرار دارند را استخراج کند و از روی آنها فایل‌ها HTML تولید کند.

۱۷. آشنایی با Spring و مفاهیم آن:

در جلسه اول از این دوره با فلسفه وجودی و نحوه معرفی و ورود Spring به دنیای جاوا آشنا می‌شوید. همچنین می‌آموزید که چگونه Spring مسائل پیچیده‌ای که تا قبل از آن راه‌حل‌های پیچیده‌ای داشت را حل کرد و به مرور توسعه یافت تا راه‌حلی برای تمام نیازهای برنامه‌ها باشد. بعد از آن با معماری و اجزای Spring آشنا می‌شوید.در بخش دوم از این جلسه یک پروژه مبتنی بر Spring را راه اندازی می‌کنیم و بخش‌های اصلی و ارتباطات آن‌ها را توضیح می‌دهیم. مهم‌ترین عناوینی که در این جلسه مطرح می‌شوند به طور خلاصه عبارت‌اند از:

- تاریخچه و کاربرد Spring
- معماری Spring
- معرفی مفاهیم Bean و Container و پیکربندی Spring
- پیاده سازی یک مثال ساده و توضیح اجزا و ارتباطات آن

Spring یک فریم ورک جاوایی است که مشابه هر فریم ورک دیگری، زیرساختی فراهم می‌کند تا برنامه‌تان را سریع‌تر و آسان‌تر تولید کنید، تست و نگهداری کنید. واضح است که بدون Spring هم می‌توان یک نرم افزار را تولید نمود اما با Spring فرایند تولید نرم افزار آسان‌تر و سریع‌تر می‌شود.

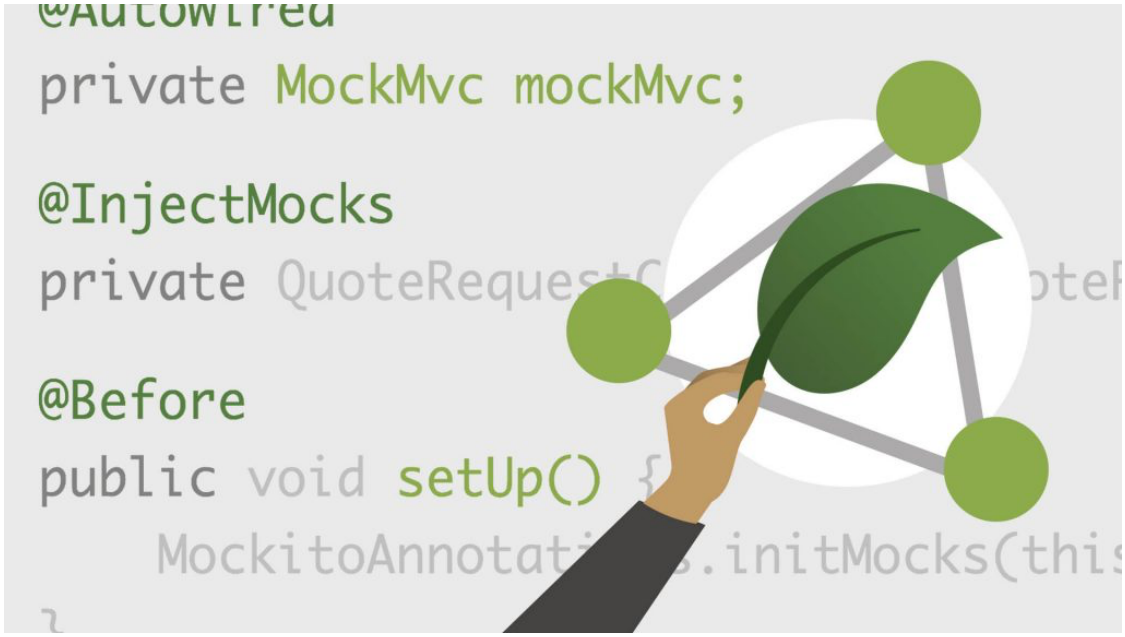
Spring چگونه تولید نرم‌افزار را تسهیل می‌کند؟

Spring طیف وسیعی از امکانات و سرویس‌ها را فراهم می‌کند و به همین دلیل می‌تواند کاربردها و کارکردهای متنوعی در معماری برنامه‌ها داشته باشد. مثلا Spring می‌تواند به عنوان زیرساختی برای تولید واسط کاربری وب استفاده شود و یا تراکنش‌های دیتابیس را مدیریت کند و یا برای یکپارچه‌سازی یک برنامه چندپارچه (Modular) استفاده شود. توجه داشته باشید که در یک برنامه مبتنی بر Spring مجبور به استفاده از همه سرویس‌های Spring نیستیم، بلکه می‌توانیم به صورت انتخابی یک یا چندین سرویسی که برای برنامه مناسب‌تر است را انتخاب کنیم. در ادامه نگاهی سریع به برخی جنبه‌های کاربرد Spring خواهیم انداخت.

معماری چندلایه:

هر نرم افزاری که تولید می‌شود از کلاس‌ها و آبجکت‌های مختلفی تشکیل شده است که در مجموع و در تعامل با یکدیگر، کارکرد آن نرم‌افزار را شکل می‌دهند. در یک نرم افزار Enterprise که سرویس‌ها متنوع و کاربران هم زیاد هستند، ممکن است تعداد این کلاس‌ها خیلی زیاد (مثلا چندین هزار) باشد. در نتیجه اگر دیاگرام کلاس که ارتباط بین کلاس‌ها و آبجکت‌های یک برنامه را نشان می‌دهد ترسیم کنیم با گرافی در هم پیچیده از آبجکت‌ها مواجه می‌شویم. اینکه هرکلاسی بتواند هرکلاس دیگری را فراخوانی کند مثل یک شهر بی‌دروپیکر می‌ماند که هرکسی هرکاری دوست داشته باشد انجام می‌دهد! تست کارکرد چنین نرم افزاری بسیار سخت خواهد بود و نگهداری آن سخت‌تر. اگر برای رفع یک اشکال در برنامه مجبور به تغییر یک کلاس شویم چه کلاس‌هایی از برنامه ممکن است تاثیرپذیرند و باید نگران کارکرد صحیح چه کلاس‌های از برنامه باشیم؟ قطعا جوابی برای این سوال وجود ندارد زیرا در یک معماری بدون چهارچوب، همه کلاس‌های برنامه بالقوه یا بالفعل ممکن است با کلاس تغییر یافته در ارتباط باشند.

تزریق وابستگی (Inversion of Control یا IOC)



مهم‌ترین عناوینی که در جلسه دوم بحث می‌شوند عبارت‌اند از:

- تعریف Bean
- محدوده حیات Beanها
- وابستگی Beanها
- تزریق مقادیر ساده (primitives)

- استفاده از علامت‌های Required , @Autowired , @Primary , @Resource , @PostConstruct , @@PreDestroy
- علامت‌های @Named , @Inject
- تنظیم و پیکربندی آبجکت‌ها با استفاده از @Import , @Configuration , @Bean
- Beanهای مبتنی بر پروفایل
- کار با رخدادهای Application Context (ایجاد و خاتمه)
- آبجکت‌های Bean Factory و Resource

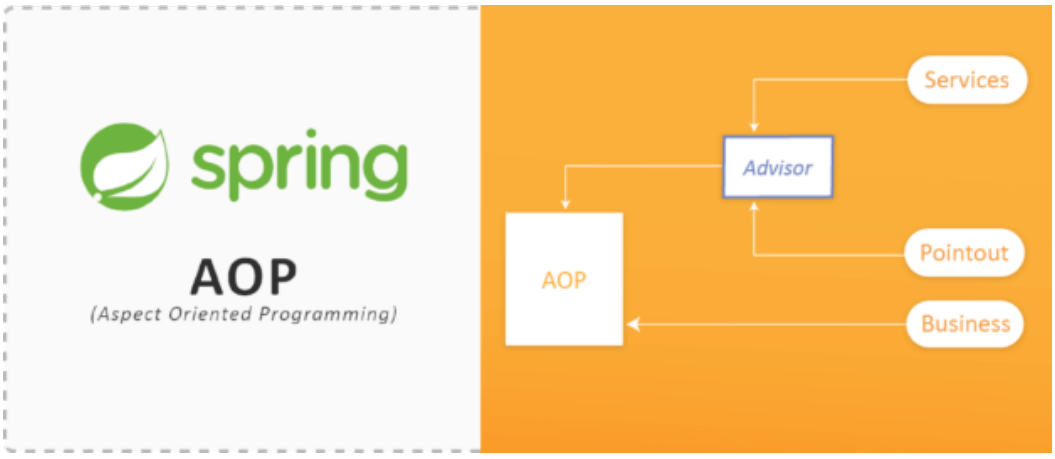
در جلسه دوم به مهم‌ترین و اصلی‌ترین قابلیت Spring یعنی IOC خواهیم پرداخت. گفته می‌شود که IOC در هسته Spring قرار دارد، این بدین معنی است که برای استفاده از IOC به هیچ کتابخانه خارجی نیاز نیست و اساسا هر پروژه‌ای که مبتنی بر Spring باشد از قابلیت IOC بهره می‌برد. وقتی یک پروژه مبتنی بر Spring را پیاده سازی می‌کنیم، دیگر آبجکت‌های برنامه را با استفاده از New ایجاد نمی‌کنیم. بلکه ایجاد آن‌ها را به عهده Spring می‌گذاریم. به عبارت دیگر در پروژه Spring آبجکت‌های برنامه را ما ایجاد نمی‌کنیم بلکه برنامه ما با فرض اینکه آبجکت‌ها ایجاد شده‌اند صرفا (با فراخوانی متدها و فیلدهای آن‌ها) از آن‌ها استفاده می‌کند. به این آبجکت‌ها که Spring آن‌ها را ایجاد می‌کند Bean گفته می‌شود.

۱۸. برنامه نویسی مبتنی بر صفت (Aspect Oriented Programming یا AOP) و ارتباط با دیتابیس (Hibernate)

مهم‌ترین موضوعاتی که در این جلسه ارائه می‌شوند عبارت‌اند از:

- مفاهیم و اجزای AOP
- معرفی آبجکت‌های Target، Advice، Advisor، Proxy و Interceptor
- عبارت‌ها و انتخاب کننده‌های Pointcut
- آشنایی با صفت‌های Before، After Returning، After Throwing، finally (After) و Around
- پشتیبانی از Aspectj

روش شی‌گرایی که امروزه در طراحی و پیاده‌سازی اغلب نرم افزارها استفاده می‌شود کامل‌ترین و پذیرفته‌ترین روش مدل کردن دنیای واقعی است. این روش، با ارائه مفاهیمی از قبیل کلاس، شی، ارث‌بری، متد و فیلد امکان مدل‌سازی دنیای واقعی را فراهم می‌کند اما فاقد مفهومی تحت عنوان «صفت» است. برای درک این نقصان به یک مثال ساده



توجه کنید. کلاسی را تصور کنید که دارای یک متد است، حال به نیازمندی‌های زیر توجه کنید:
هدف Aspect Oriented Programming که به آن به اختصار AOP یا «برنامه نویسی مبتنی بر صفت» گفته می‌شود حل مسئله فوق است.

مهم‌ترین عناوینی که در این جلسه ارائه می‌شوند عبارت‌اند از:

- نگاشت شی‌گرایی به رابطه‌ای(Object Relational Mapping)
- تنظیم و پیکربندیHibernate
- نگاشت آبجکت‌های جاوا به جداول دیتابیس
- نگاشت ارث‌بری(Inheritance)
- نگاشت وابستگی(Association)
- مقایسه Lazy وEager
- کوئری روی دیتابیس با استفاده از HQL و Criteria

تقریبا با قطعیت می‌توان گفت تمام برنامه‌های اینترپرایز از پایگاه داده استفاده می‌کنند. پایگاه داده‌ها خود برنامه‌هایی هستند که کار ذخیره‌سازی و بازیابی داده‌ها را بر عهده دارند و انواع و اقسامی دارند که معروف‌ترین آن‌ها دیتابیس‌های رابطه‌ای (Relational Databases) و دیتابیس‌های غیررابطه‌ای (NoSQL Databases) هستند.
گرچه از معرفی و ارایه دیتابیس‌های غیررابطه‌ای زمان زیادی می‌گذرد. همچنان دیتابیس‌های رابطه‌ای پرطرفدارتر و پراستفاده‌تر هستند. شاید عمده دلیل این محبوبیت، سابقه زیاد این دیتابیس‌ها و دانش و تجربه زیادی است که در حوزه این دیتابیس‌ها وجود دارد.
.در این جلسه با نگاهی با تاریخچه Hibernate معماری این فریم‌ورک را بررسی می‌کنیم. نحوه پیکربندی و استفاده از آن را با یک برنامه ساده ملاحظه خواهیم کرد و در ادامه جزییات نگاشت کلاس‌های برنامه (Entity) به جداول دیتابیس (Table) را بحث می‌کنیم. موضوعات دیگری از قبیل نحوه نگاشت ارتباط بین آبجکت‌ها (Association) و ارث‌بری (Inheritance)، نوشتن کوئری مبتنی بر آبجت (HQL و Criteria) از مباحث دیگر این جلسه هستند.

۱۹. یکپارچه‌سازی Spring و Spring Data (Hibernate)

مهم‌ترین موضوعاتی که در این جلسه ارائه می‌شوند عبارت‌اند از:

- معرفی آبجکت‌هایRepository
- فعال سازی Repositoryها
- مدیریت تراکنش
- استفاده از کوئری‌های پارامتریک
- Projection
- Locking

اگرچه استفاده از Hibernate کار با دیتابیس‌ها را تا حد زیادی آسان می‌کند اما استفاده از Hibernate آنقدرها هم بی‌دردسر نیست. Spring Data که یکی دیگر از پروژه‌های اکوسیستم Spring است. با ارایه یک لایه روی Hibernate تلاش کرده است تا حد امکان از مشکلات و مسایل کار با Hibernate بکاهد. Spring Data اولا با ارایه یک API غنی،

پیاده سازی کلاس‌های لایه دیتا را بسیار ساده می‌کند و نیاز به کوئری نویسی را (به جز موارد خاص) از بین می‌برد. ثانیا مدیریت تراکنش را (که در Hibernate کار پرزحمتی است) را با استفاده از Spring AOP تسهیل می‌کند.

۲۰. برنامه نویسی وب (Restful Services) و Spring Security

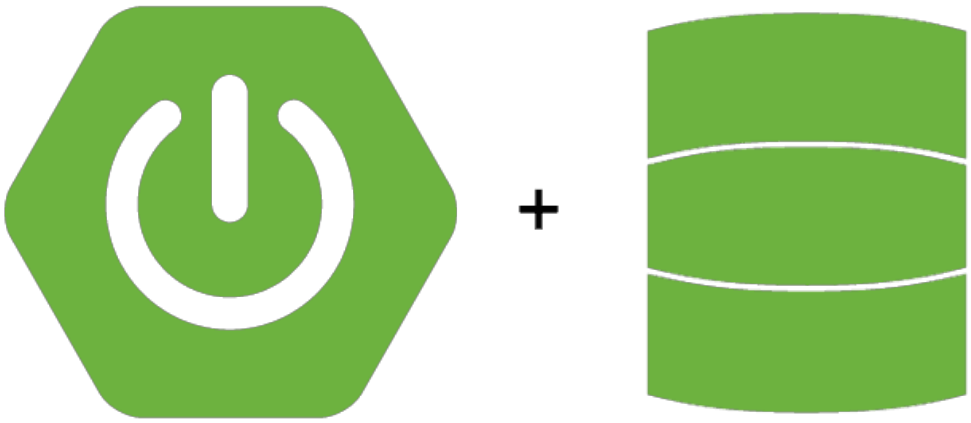
در جلسه ششم از دوره Spring، نحوه پیاده سازی سرویس‌های نوع REST با استفاده از فریم ورک Spring خواهیم پرداخت. در ادامه این جلسه موضوع مدیریت خطاهای سرویس و برخی دیگر از موضوعات مهم در رابطه با سرویس‌های REST از قبیل مدیریت نسخه‌ها، صفحه بندی و مرتب کردن نتایج سرویس نیز مطرح می‌شوند. در پایان نحوه مستندسازی این سرویس‌ها نیز مطرح و بحث می‌شوند.

مهم‌ترین موضوعاتی که در این جلسه ارائه می‌شوند عبارت‌اند از:

- معماری RESTful
- پیاده سازی سرویس‌های RESTful درSpring
- مدیریت خطاها
- مستندسازی سرویس‌هایRESTful
- مدیریت نسخه‌ها(Versioning) ، مرتب سازی(Sorting) ، و صفحه بندی(Paging)

تکنولوژی وب، تکنولوژی است که برای ارتباط بین مرورگر وب و وب-سرور استفاده می‌شود و براساس پروتکل‌های رایجی از قبیل HTTP و HTTPS کار می‌کند. در این پروتکل‌ها، داده‌های متنی که عموما هم HTML هستند بین مرورگر وب و وب-سرور ارسال و دریافت می‌شوند. نکته ای که در تکنولوژی وب نهفته است این است که مرورگر وب و وب-سرور هر کدام ممکن است به هر زبانی نوشته شده باشد و روی هر پلتفرمی(Windows ، Linux، Solaris، ...) اجرا شود، اما از آنجایی که با پروتکل‌های شناخته شده‌ای کار می‌کنند که مستقل از هر پلتفرم و زبانی هستند و فرمت داده‌های آن‌ها نیز متن است که آن هم مستقل از هر زبان و هر پلتفرمی است آن‌ها می‌توانند با هم ارتباط داشته باشند. بر همین مبنا یک تکنولوژی جدید معرفی شده است که به آن وب-سرویس گفته می‌شود که مبتنی برتکنولوژی وب است و امکان

می‌دهد تا نرم افزارها با استفاده از پروتکل‌های وب (HTTP و HTTPS) و با ارسال و دریافت متنXML) ، JSONیا متن ساده (با یکدیگر یکپارچه شوند



.

تکنولوژی وب-سرویس در دو نوع اصلی SOAP و REST معرفی شده است، نوع SOAP که قدیمی تر است مبتنی بر XMLاست اما به علت سربار که دارد) حجم زیادی دارد و ترافیک زیادی از شبکه را اشغال می‌کند، تولید و پردازش آن پرهزینه است، امکان Cache کردن و Scale کردن آن وجود ندارد (به مرور به حاشیه رفته و نوع REST دقیقا به همان دلایلی که نوع SOAP از آن بی‌بهره است پرطرفدار شده است.
وب-سرویس نوع REST به مرور زمان و با ارایه معماری میکروسرویس بسیار پرطرفدار شده است. در معماری میکروسرویس، اجزای برنامه به سرویس‌های مستقل و جزیی تجزیه می‌شوند که همگی با استفاده از سرویس‌های REST با یکدیگر تعامل دارند.

مهم‌ترین موضوعاتی که در این جلسه بحث می‌شوند عبارت‌اند از:

- پیکربندی و تنظیمات
- استفاده از دیتابیس
- استفاده از JDBC، JPA (Hibernate) برای شناسایی کاربران
- استفاده از Encoderها
- فعالسازی HTTPS
- استفاده از تگ‌های Spring Security

Spring Security یکی از پروژه‌های اکوسیستم Spring است که با بهره‌گیری از سرویس‌هایی که Spring فراهم می‌کند (از قبیل IOC و AOP) بستری را برای توسعه نیازمندی‌های امنیتی یک پروژه جاوایی فراهم می‌کند. واضح است که اصلی‌ترین نیازمندی امنیتی هر سیستم نرم‌افزاری «شناسایی کاربران» (Authentication) و «کنترل دسترسی کاربران» (Authorization) است. یک کاربر می‌تواند یک کاربر انسانی باشد که کار با نرم افزار را دارد یا ممکن است یک نرم افزار یا سیستم دیگر باشد که با سیستم ما در تعامل است. بنابراین ما از زیرساخت امنیت، انتظار داریم تا با نام کاربری و رمز عبوری که وارد می‌کنند آن‌ها را تایید یا رد کند. همچنین وقتی یک کاربری تایید شد از زیرساخت امنیت سیستم انتظار داریم تا دسترسی وی به منابع سیستم را کنترل کند. به عبارت دیگر کنترل کند که آیا کاربر تایید شده حق دسترسی به یک منبع درخواست کرده را دارد یا ندارد. علاوه بر این‌ها ممکن است بخواهیم نرم افزارمان را از طریق یکپارچه سازی

با سیستم‌های دیگر از قبیل LDAP، OpenID یا JAAS امن کنیم. به تمام این نیازمندی‌ها باید مقاومت در برای نفوذپذیری‌های رایج از قبیل CSRF یا Session Hijacking را نیز اضافه کنیم. این‌ها سرویس‌های متنوع و جامعی هستند که توسط Spring Security فراهم می‌شوند.

از مدت‌ها قبل از Spring Security، پلتفرم جاوا بسیاری از خدمات فوق را فراهم می‌کرد و می‌کند اما چرا با وجود اینکه جاوا سرویس امنیت را به شکل استاندارد و جهانی فراهم می‌کند، Spring Security متولد شد و هر روز پرطرفدارتر می‌شود؟ شاید ریشه اصلی محبوبیت Spring Security به محبوبیت Spring برگردد. با فراگیرشدن و محبوبیت فراگیر استفاده Spring، تمام محصولاتی که معماری مبتنی بر Spring داشته‌اند نیز شهرت و طرفدار پیدا کرده‌اند. اما در عین حال نباید قابلیت‌های بی نظیر و انعطاف پذیری فراوان Spring Security را از قلم انداخت. Spring Security با فراهم کردن طیف وسیعی از سرویس‌ها و قابلیت‌ها، تمام نیازمندی‌های امنیتی یک پروژه جاوایی را پوشش می‌دهد. پیکربندی و تنظیمات Spring Security می‌تواند کاملا به صورت تعریفی (استفاده از فایل XML) انجام شود که نگهداری آن را بسیار ساده می‌کند. بسیاری از خصوصیات Spring Security قابل توسعه است و بنابراین بسته به نیازمندی‌های امنیتی خاص یک پروژه امکان تغییر و بومی سازی اجزای آن وجود دارد. قابلیت حمل دارد یعنی وقتی در یک پروژه تنظیم و پیکربندی شد بدون توجه به محیط اجرایی و بدون نیاز به پیکربندی محیط اجرایی همراه با پروژه منتقل می‌شود. همچنین باید گفت بسیاری از قابلیت‌هایی که Spring Security فراهم می‌کند در پلتفرم Java EE وجود ندارد و تا اضافه شدن به این پلتفرم ممکن است زمان زیادی بگذرد. این‌ها دلایل کافی هستند که ما Spring Security را به عنوان بستر و زیربنای امنیتی پروژه‌مان انتخاب کنیم.

۲۱. Test with Spring و Spring Batch

در طول تولید یک نرم افزار، انواع مختلفی از تست‌ها انجام می‌شود، تست عملکرد و تست کیفیت از جمله این تست‌ها هستند، اما آنچه موضوع این جلسه است Unit Test و Integration Test هستند. منظور ازUnit ، کوچکترین جزء یک برنامه است و Unit Test به تستی گفته می‌شود که روی کوچکترین قسمت از یک برنامه یعنی متد انجام می‌شود. و منظور از Integration Test (یکپارچگی) تستی است که یکپارچگی اجزای برنامه را تایید می‌کند.

برای این دو نوع تست، کدنویسی می‌شود، یعنی برنامه نویسان همچنان که منطق برنامه را پیاده سازی می‌کنند، کدهایی برای تست جزء و تست یکپارچگی نیز پیاده سازی می‌کنند. دلیل آن هم این است که اولا این دو نوع تست ذاتاً مرتبط با کدهای برنامه و منطق برنامه هستند، ثانیا، با وجود کدهای تست، می‌توان به تست اتوماتیک رسید زیرا با هربار Build برنامه به صورت اتوماتیک اجرا می‌شوند.
برای پیاده سازی Unit Test دو فریم ورک محبوب JUnit و TestNG در جاوا وجود دارند. تنوع فریم ورک‌ها برای پیاده سازی تست‌های یکپارچگی بسیار بیشتر است که یکی از محبوب‌ترین آن‌ها Mockito است که در این جلسه به آن خواهیم پرداخت.

برای تست پروژه‌های مبتنی برSpring ، علاوه بر داشتن دانش تست و مهارت کار با فریم ورک‌های تست، لازم است معماری Spring را خوب درک کرده باشید و نحوه تعامل اجزای یک پروژه Spring را بدانید. با همه این‌ها بازهم تست پروژه‌هایSpring ای پیچیدگی‌ها و مسائل خودشان را دارند، به همین دلیل Spring تلاش کرده است تا با ارایه فریم ورک تست، پیاده سازی تست در پروژه‌هایSpring ای را تا حد امکان ساده کند. در این جلسه علاوه بر آشنایی با مفاهیم تست جز و یکپارچگی با فریم ورک‌های تست JUnit و Mockito آشنا می‌شوید و در انتها نحوه تست پروژه‌های مبتنی بر Spring را خواهید آموخت.

مهم‌ترین موضوعاتی که در این جلسه بحث می‌شوند عبارت‌اند از:

- پردازش دسته‌ای چیست؟
- معرفی اجزای اصلی پردازش دسته‌ای(Reader ، Processorو(Writer
- معرفی مفاهیم و بازیگران پردازش دسته‌ای
- تنظیم، پیکربندی و اجرای فرایند دسته‌ای
- زمان‌بندی اجرای پردازش دسته‌ای

Spring Boot .۲۲

مهم‌ترین موضوعاتی که در این جلسه ارائه می‌شوند عبارت‌اند از:

- Spring Boot.چیست
- معرفیStarter ها
- پیکربندی و اجرای پروژهSpring Boot
- الصاق تنظیمات برنامه و آبجکت‌های برنامه
- تعریف تنظیمات برنامه در فایل‌های Properties وYaml
- پروفایل درSpring Boot
- مانیتورینگ برنامه Spring Boot با استفاده ازActuator
- تولید لاگ
- تست

همانطور که در جلسه اول مطرح شد، Springیک فریم ورک همه منظوره است که طیف وسیعی از ویژگی‌ها و قابلیت‌ها را داراست. همین موضوع باعث شده که پروژه‌های مبتنی بر Spring پیچیده و پیکربندی آن دشوار باشد. به همین دلیل پروژه Spring Boot ابداع شده است تا با ارایه تنظیمات پیش فرض، که به آن Convention over Configuration گفته می‌شود کار تنظیم و راه اندازی پروژه‌های مبتنی بر Spring را تا حد زیادی آسان و ساده کند.
بسته به اینکه از کدامیک از سرویس‌های Spring قصد استفاده داشته باشید، در Spring Boot یک starter ارائه شده است که وابستگی‌ها (dependency) و تنظیمات پیش فرض مربوط به آن سرویس را شامل می‌شود. بنابراین صرفا با افزودن یک starter می‌توان یک سرویس Spring) مثلا ارتباط با دیتابیس، تولید سرویسREST ، Spring Security) را به پروژه اضافه کرد.

در این جلسه، مفهوم Profile را خواهید آموخت که نقش کلیدی در پیاده سازی پروژه‌های Spring Boot دارد و امکان می‌دهد یک پروژه را برای محیط‌های مختلف (محیط توسعه، تست، عملیاتی، استیجینگ) تنظیم و پیکربندی کرد.
در ادامه این جلسه نحوه مانیتورینگ سلامت پروژه (Actuator) را خواهید آموخت. واضح است وقتی پروژه‌ای در محیط عملیاتی قرار می‌گیرد، پیگیری وضعیت اجرای برنامه و سلامت سرویس‌های برنامه بسیار حیاتی است، به عنوان نمونه وقتی یک سرویس به دلایل نامشخصی خطا تولید می‌کند لازم است قبل از آنکه مشتری آن سرویس به ما وجود خطا را اطلاع دهد، لازم داریم تا از وجود خطا در آن سرویس مطلع شویم.
از موضوعات مهمی که در این جلسه مطرح می‌شوند، بحث لاگ و نحوه تولید لاگ و پیاده سازی تست در برنامه Springای است.

مدرس این دوره احمدرضا صدیقی کیست؟

- معمار ارشد در حوزه جاوا مربوط به پروژه دانشگاه علوم پزشکی
- معمار ارشد در حوزه جاوا مربوط به پروژه شرکت خبره پردا
- معمار ارشد در حوزه جاوا مربوط به پروژه شرکت کیاتک بنیا
- معمار ارشد در حوزه جاوا مربوط به پروژه دانشگاه مالک اشتر
- مشاور پروژه ملی طرح جامع مالیاتی
- مشاور پروژه ملی وزارت بهداشت
- مشاور پروژه بانک ملت
- مولف مجموعه کتاب‌های جاوا (فارسی و انگلیسی)
- بیش از ۱۲ سال سابقه تدریس جاوا
- ارائه فریم‌ورک تخصصی جاوا (اطلس

نحوه دریافت دوره‌های دانلودی چگونه است؟

پس از ثبت سفارش، به حساب کاربری خود در سایت نیک‌آموز وارد شده و در بخش «**دانلودها**» اقدام به دانلود جلسات دوره خریداری شده کنید.

صدور فاکتور رسمی چگونه است؟

در صورت تمایل به دریافت فاکتور رسمی، پیش از خرید خود با واحد فروش مجموعه (۰۲۱ – ۹۱ ۰۷ ۰۰ ۱۷) تماس حاصل نمایید.**شایان ذکر است، امکان صدور فاکتور رسمی پس از خرید آنلاین از سایت مجموعه به هیچ عنوان وجود نخواهد داشت.**

پشتیبانی بی‌نظیر

آموزش بدون پشتیبانی کاملا بی‌معنی است، الان تمام دوره‌های نیک آموز دارای پشتیبانی از طریق سایت و تلگرام است.

پس ثبت و نهایی شدن سفارش شما در سایت نیک‌آموز، تیم پشتیبانی طی ۲۴ تا ۷۲ ساعت کاری با شما تماس خواهند گرفت تا فرایند عضو شدن شما در گروه پشتیبانی تلگرامی هر دوره انجام شود.در صورت وجود هر گونه سوال و ابهامی می‌توانید با شماره‌های شرکت تماس حاصل فرمایید و یا از طریق بخش چت پشتیبانی سایت، سوالات خود را مطرح نمایید.

آدرس: تهران، یوسف آباد، میدان فرهنگ، خیابان ۳۳، پلاک ۲۹، زنگ ۲، دفتر نیک آموز

شماره تماس: ۱۷ ۰۰ ۰۷ ۹۱ – ۰۲۱ | **موبایل فروش:** ۰۹۱ ۴۰۰۶ ۲۰۶