FEATURED COURSES

Clean Code Kata

> “ Any fool can write code that a computer can understand. Good programmers write code that humans can understand. ”
>
> — *Martin Fowler*

Clean Code is the practice of writing code that is readable, simple, elegant, and therefore easy to maintain. Code that is clean does not turn against the developer, rather it helps to better understand the domain of interest and the solution to be adopted. Unfortunately, it is often easier said than done.

"Kata" is a Japanese term that refers to exercises to be repeated constantly to improve oneself.

This course begins with a description of all the practices collected by Robert C. Martin in the book titled "Clean Code" and continues with three Katas, or exercises of increasing difficulty in which the acquired knowledge is put into practice, along with software engineering concepts and the application of refactoring and test-driven development.

Prerequisites

The course is aimed at software developers who know at least one object-oriented language.

Duration

The expected duration of the course is 16 hours.

Material

The slides used for the theoretical part and the code that implements the solution to the exercises are available.

Program

1. Theory
    1. Clean Code
    2. Katas
2. Practice
    1. Analysis and refactoring of non-clean code
    2. Comparative analysis on the flexibility of four different solutions to solve a problem
    3. Solution of a complex problem through the implementation of simple and reusable modules

Inglorious Coderz di Mistretta Matteo Antony
Via Val Della Torre, 39
10149 Torino (TO)

Hangouts: antonymistretta
Skype: antony.mistretta

antony.mistretta@gmail.com
www.ingloriouscoderz.it

Test-Driven Development / Behaviour-Driven Development

> “ Code never lies, comments sometimes do. ”
>
> — *Ron Jeffries*

One of the most controversial practices promoted by the Agile methodology called Extreme Programming is precisely the Test-Driven Development: the customer pays me to write code and wants it ready for tomorrow, and I am supposed to waste time writing test code that was not requested nor will be paid? The answer is yes. This is because Test-Driven Development (abbreviated TDD), although at the beginning slows down development, at steady helps to keep the code stable, simple and even well documented, with great joy both of the developer and of the Customer.

In this course TDD is exposed in the context of an Agile process, showing its short and long term benefits, then Behavior-Driven Development (BDD) is illustrated as an evolution of TDD, and the applicability of these concepts in the business context of the class is discussed. Optionally, practical TDD / BDD exercises are performed in a programming language of choice.

Prerequisites

Anyone working in IT can benefit from the theoretical part: project managers, IT managers, programmers. For the practical part some programming knowledge is required.

Duration

The expected duration is 4 hours for the theoretical part, plus another 4 hours for the practical part.

Material

The slides used during the course and the source code for the practical part are available.

Program

1. What is TDD
2. The advantages of TDD
3. What is BDD
4. TDD / BDD exercises

Inglorious Coderz di Mistretta Matteo Antony
Via Val Della Torre, 39
10149 Torino (TO)

Hangouts: antonymistretta
Skype: antony.mistretta

antony.mistretta@gmail.com
www.ingloriouscoderz.it

Extreme Programming

> “ I'm not a great programmer; I'm just a good programmer with great habits. ”
>
> — *Kent Beck*

Among all Agile methodologies, Extreme Programming is certainly the most misunderstood. Yet nowadays many of its practices, such as Test-Driven Development or Continuous Integration, are on the agenda of every modern IT company. Extreme Programming (abbreviated XP) is complementary to other Agile methodologies: Scrum, for example, deals with high level project management, while XP applies to the production of code and its release in production.

In this theoretical course, we describe the activities involved in XP, the underlying values and the practices that derive from it, and we discuss the applicability of these concepts in the business context of the class.

Prerequisites

The only prerequisite is a basic knowledge of what it means to develop software, so programmers are welcome but also project managers and team leaders.

Duration

The expected duration is 16 hours.

Material

The slides used during the course are available.

Program

1. Activities
    1. Coding
    2. Testing
    3. Listening
    4. Designing
2. Values
    1. Communication
    2. Simplicity
    3. Feedback
    4. Courage
    5. Respect
3. Practices
    1. Pair Programming
    2. Planning Game
    3. Test-Driven Development
    4. Whole Team
    5. Continuous Integration
    6. Design Improvement
    7. Small Releases
    8. Standard Coding
    9. Collective Code Ownership
    10. Simple Design
    11. System Metaphor
    12. Sustainable Peace

Inglorious Coderz di Mistretta Matteo Antony
Via Val Della Torre, 39
10149 Torino (TO)

Hangouts: antonymistretta
Skype: antony.mistretta

antony.mistretta@gmail.com
www.ingloriouscoderz.it

Web 2.0

> “ The world is changed. I feel it in the water. I feel it in the earth. I smell it in the air. ”
>
> — *Galadriel, Lord Of The Rings*

The world is changing. And with it the web, which evolves too fast to remain anchored to legacy software and old-fashioned mentality.

In this theoretical course, an overview of the most recent technologies and methodologies that have emerged in the international technological scenario, from network architectures to the most innovative frameworks to cloud solutions, is shown. The lesson is accompanied by various practical demonstrations.

Prerequisites

The course can be followed without problems by technical managers, project managers, systems engineers and developers.

Duration

The expected duration is 8 hours.

Material

The slides used during the course are available, in which there are numerous in-depth links.

Program

1. Architectures
    1. Evolution of software architectures
    2. Application Server vs. Microservices
    3. REST
    4. REST-based protocols
    5. GraphQL
2. Back End
    1. Relational DBMS vs. NoSQL
    2. Web Framework: Evolution vs. Revolution
    3. Server: Multithread vs. Event-Driven
3. Front End
    1. Evolution of the web
    2. JavaScript framework
    3. CSS Framework
    4. Furniture evolution
    5. App native
    6. Compile-to-native app
    7. Hybrid app
    8. Progressive Web Apps
4. Agile
    1. Introduction to Extreme Programming
    2. Small Releases
    3. Test-Driven Development
    4. Refactoring
    5. Standard Coding
    6. Continuous Integration
5. Deploy
    1. Git
    2. CI / CDE / CD software
    3. Virtual Machine
    4. Container
    5. Orchestrator
    6. Cloud / PaaS / Serverless

Inglorious Coderz di Mistretta Matteo Antony
Via Val Della Torre, 39
10149 Torino (TO)

Hangouts: antonymistretta
Skype: antony.mistretta

antony.mistretta@gmail.com
www.ingloriouscoderz.it

Design Patterns

"Reinventing the wheel" is an Anglo-Saxon idiom expression used when a generally accepted technical solution is ignored in favor of locally recreated solutions. Many software problems can instead be traced back to a common root that can be solved with a specific solution. Design patterns are standard solutions to recurring problems, so they are a great help to find a solution, communicate it effectively, and above all avoid reinventing the wheel.

In this course we define the design patterns in general and show an overview of the best known design patterns, comparing them with the business context of the class. We also expose some anti-patterns, useful to avoid common mistakes during software design.

Prerequisites

Although code writing is not intended, solid object-oriented programming bases are still required.

Duration

The expected duration is 8 hours.

Material

The slides used during the course are available.

Program

1. Code smell
2. What is a design pattern
3. GRASP
    1. Controller
    2. Creator
    3. High Coesion
    4. Information Expert
    5. Low Coupling
    6. Polymorphism
    7. Protected Variations
    8. Pure Fabrication
    9. Indirection
4. The Gang Of Four patterns
    1. Abstract Factory / Builder / Factory Method
    2. Prototype / Flyweight
    3. Singleton
    4. Adapter
    5. Composite
    6. Decorator / Proxy
    7. Command
    8. Mediator
    9. Memento
    10. Observer / Pub-Sub
    11. State / Strategy
5. The SOLID principles
    1. Single Responsibility
    2. Open / Closed
    3. Liskov Substitution
    4. Interface Segregation
    5. Dependency Inversion
6. Anti-patterns
    1. Anemic Domain Model
    2. God Object
    3. Spaghetti Code
    4. Premature Optimization

Inglorious Coderz di Mistretta Matteo Antony
Via Val Della Torre, 39
10149 Torino (TO)

Hangouts: antonymistretta
Skype: antony.mistretta

antony.mistretta@gmail.com
www.ingloriouscoderz.it

Java Web Frameworks

The Java language is perhaps the main reason why object-oriented programming exploded in the 1990s. Since then, the language has evolved, as well as the technologies based on it, but unfortunately not without errors: for a long time technologies like the J2EE platform have introduced and even suggested solutions that are all too complex and heavy. To counter this *over-engineering* in the 2000s, frameworks such as Struts, Spring and Hibernate have used and encouraged in the proper wat those design patterns that are fundamental to good design, thus enabling developers to build enterprise solutions by simply focusing on modeling the domain of interest.

In this course, very practical but supported by different software engineering concepts, we start from the creation of a simple web application with CRUD functionality built with servlets and JSPs and then progressively analyze the difficulties involved in scaling the application. In this way the Java frameworks and the underlying theoretical principles are introduced one by one, up to a complete web application that is also simple, robust and flexible.

Prerequisites

The course is intended for developers who have a good knowledge of Java and object-oriented programming.

Duration

The expected duration is 48 hours.

Material

The slides used during the course are available, which include in-depth links, plus the source code related to the tutorials.

Program

1. Maven
    1. Use for command line applications
    2. Use for web applications
    3. Resolution of dependencies
    4. Automation of the compilation and deployment process
2. Servlets and JSPs
    1. Creation of a web app with CRUD features
    2. The Decorator / Proxy design pattern
    3. Filter
3. Struts 2
    1. The Observer design pattern
    2. The MVC architectural pattern
    3. Migration of the web app from Servlets and JSPs to Struts
4. Spring
    1. Singleton as an anti-pattern
    2. The Inversion Of Control principle
    3. The design pattern Dependency Injection
    4. Migration of the web app to Spring
    5. Integration between Spring and Struts
    6. Aspect-Oriented Programming
    7. Spring Web
    8. Migration of the web app from Struts to Spring Web
    9. Spring MVC
    10. Creation of REST API with Spring MVC
5. Hibernate
    1. Object-Relational Mapping
    2. Migration of the web app to Hibernate
    3. Integration between Spring and Hibernate
    4. Representation of relations between tables
    5. Creation of a Facebook-like web app

Inglorious Coderz di Mistretta Matteo Antony
Via Val Della Torre, 39
10149 Torino (TO)

Hangouts: antonymistretta
Skype: antony.mistretta

antony.mistretta@gmail.com
www.ingloriouscoderz.it

ESNext

One of the languages that is having the greatest impact on today's technology is JavaScript (not to be confused with Java): from a simple scripting language on the browser it became the lingua franca to create ultra-high performance servers, desktop/mobile/web applications, and it can even be used on embedded devices and virtual reality environments. Its standardization by ECMA International (which renamed it ECMAScript, or simply ES), and the innovations brought by the big web players such as Google, Microsoft, Facebook and Mozilla, have quickly brought it to the top of the most popular technologies ranking on StackOverflow.

Unfortunately for many the evolution of ES is happening a little too quickly: the language adds new syntax elements every year, started to embrace different programming paradigms, and its ecosystem is so fervent that it is said that not a day goes by without a new framework being born. It is therefore necessary to extricate oneself in this chaos, carefully selecting the really relevant information and discarding those that are more secondary or close to disappearing in this race for the perfect web technology.

In this course we take a look at the history of ECMAScript, the problems that have been solved, the new opportunities offered by the new ecosystem and the challenges yet to be overcome. Furthermore, some of the new functionalities of the language are tested through a practical exercise.

Prerequisites

To successfully follow this course it would be useful to have some programming basics in any language.

Duration

The expected duration is 16 hours.

Material

The slides used during the course are available, plus the source code for the practical part.

Program

1. The JavaScript development environment
    1. Node and NPM
    2. Express
2. New features introduced by ES2015
    1. Synchronous and asynchronous modules
    2. New syntax: template strings, const and let, arrow functions
    3. Destructuring and spread operator
    4. Functional JS (map, reduce, currying)
    5. Fetch API
    6. Promises, generator functions, async / await
    7. Object-oriented JS (classes, patterns)
    8. Babel, ESLint, Typescript, Flow, Webpack
3. Transpilation
    1. History: CoffeeScript
    2. TypeScript
    3. Babel/JSX/ESNext
    4. Less/Sass/Stylus
    5. Jade/Pug
4. Automation
    1. History: Grunt/Gulp/Yeoman
    2. Webpack
5. Test
    1. BDD and Jasmine/Jest
6. Frameworks
    1. History: Backbone/Knockout/AngularJS
    2. React
    3. Angular
    4. Vue.js

Inglorious Coderz di Mistretta Matteo Antony
Via Val Della Torre, 39
10149 Torino (TO)

Hangouts: antonymistretta
Skype: antony.mistretta

antony.mistretta@gmail.com
www.ingloriouscoderz.it

JavaScript Frameworks

The rapid evolution of the JavaScript language, and the interest shown by the big players of the web such as Google, Microsoft and Facebook, has led to a plethora of competing frameworks. Among these the winners, that is those who have finally become more established, are three:

- Angular, highly appreciated in the enterprise for its similarity with other frameworks used on the back end and for being supported by companies such as Google and Microsoft;
- React, which has a slightly higher learning curve but allows for more expressivity with less code and optimal performance; and
- Vue.js, which manages to combine the simplicity of use of Angular, the performance of React, and the elegance we want to achieve one day with web components.

In this course we try to implement the same CRUD application with all three frameworks, making a comparative analysis of the pros and cons of each. In addition, we discuss which framework best suits the needs of the customer and the objectives that are set.

Prerequisites

Basic programming knowledge in any language is required. It is also preferable to have followed the ESNext course.

Duration

The expected duration is 24 hours.

Material

The slides used during the course are available, plus the source code for the practical part.

Program

1. Angular
    1. Features of the framework
    2. Creation of a web app
2. React
    1. Framework or library?
    2. Features of the framework
    3. Implementation of a web app
3. Vue.js
    1. Features of the framework
    2. Creation of a web app

Inglorious Coderz di Mistretta Matteo Antony
Via Val Della Torre, 39
10149 Torino (TO)

Hangouts: antonymistretta
Skype: antony.mistretta

antony.mistretta@gmail.com
www.ingloriouscoderz.it

Angular

Angular is the most used framework in the enterprise, due to its similarity with other frameworks used on the back end and because it is developed and sponsored by Google in collaboration with Microsoft. Its learning curve is relatively low if you know the concepts of object-oriented programming and the patterns used in the back-end like Dependency Injection.

Recently, Angular has incorporated some concepts and technologies borrowed from other areas, such as Redux and RxJS. In this case functional programming comes into play and things become more interesting.

In this course a web application is created from scratch with CRUD functionality and access to an external server, illustrating gradually the characteristics of the framework.

Prerequisites

Knowledge of at least one object-oriented language is required. It is also preferable to have followed the ESNext course.

Duration

The expected duration is 40 hours.

Material

1. From AngularJS to Angular
2. Framework architecture
3. Data binding
4. Components
    1. Hierarchy of components
    2. Communication between components
    3. Component lifecycle
    4. Updates and render
5. Services
6. Dependency Injection
7. Integration with RxJS
    1. Functional management of asynchronous flows
    2. RxJS
    3. Management of Angular events with Observable and Subject
8. Integration with Redux
    1. Basics of functional programming
    2. The three principles of Redux
    3. Action types, action creators, store.dispatch
    4. Reducers, selectors, store.subscribe
    5. Behavior-driven development with Jest
    6. ngrx
    7. Middlewares and reduxDevTools
    8. Asynchronous calls with ngrx-effects
9. Filters
10. Projections
11. Performance optimization
    1. Cycles
        2. Immutability
        3. Debouncing
        4. Minimizing re-renders with ChangeDetectionStrategy.OnPush

Inglorious Coderz di Mistretta Matteo Antony
Via Val Della Torre, 39
10149 Torino (TO)

Hangouts: antonymistretta
Skype: antony.mistretta

antony.mistretta@gmail.com
www.ingloriouscoderz.it

React

React claims to be simply a library to create graphical interfaces. In reality this software developed and sponsored by Facebook is a framework to all effects, which allows one to create even large applications with incomparable performance, thanks to the ability to write simple and modular code.

The learning curve is a little higher because functional programming concepts come into play, but the basic rules to be learned are a few and everything else is logically self-contained. If you do not let it scare you, you can learn concepts that are used to fully understand any other modern technology. Learning React means learning to code well.

In this course a web application is created from scratch with CRUD functionality and access to an external server, illustrating gradually the characteristics of the framework.

Prerequisites

Basic programming knowledge in any language is required. It is also preferable to have followed the ESNext course.

Duration

The expected duration is 40 hours.

Material

The slides used during the course are available, plus the source code for the practical part.

Program

1. React
    1. create-react-app
    2. JSX
    3. Class components: props, state, instance attributes
    4. Hierarchy of components and communication between components
    5. Component lifecycle and performance optimization with shouldComponentUpdate
    6. Functional components
    7. PureComponent/React.memo
    8. Communication with the server
    9. Container component and Presentational component
    10. Forms: controlled vs. uncontrolled components
    11. Patterns: Higher Order Components and Render props/Function as child component
2. React Router
    1. Development of applications with multiple screens and creation of navigation menus
    2. Management of nested routes
3. Inline styles vs. CSS modules
4. Recompose
    1. The compose function
    2. withState, withReducer, withProps, withHandlers, lifecycle
5. Redux
    1. The three principles
    2. createStore, middlewares and reduxDevTools
    3. Action types, action creators, store.dispatch
    4. Reducers, selectors, store.subscribe
    5. Behavior-driven development with Jest
6. Reselect
7. React-Redux
8. Integration of asynchronous flows with Redux
    1. Redux-thunk
    2. Redux-saga
9. Context API
10. Hooks

Inglorious Coderz di Mistretta Matteo Antony
Via Val Della Torre, 39
10149 Torino (TO)

Hangouts: antonymistretta
Skype: antony.mistretta

antony.mistretta@gmail.com
www.ingloriouscoderz.it

Vue.js

Vue.js is a framework that is becoming very popular because it manages to combine the simplicity of use of Angular, the performance of React and the elegance of Polymer. The famous PHP framework Laravel has chosen it as the default framework for the front end, probably because it shares the linear learning curve. Knowing Vue.js means learning all the concepts that the new generation of frameworks bring with them, but starting off on the right foot.

In this course a web application is created from scratch with CRUD functionality and access to an external server, illustrating gradually the characteristics of the framework.

Prerequisites

Basic programming knowledge in any language is required. It is also preferable to have followed the ESNext course.

Duration

The expected duration is 40 hours.

Material

The slides used during the course are available, plus the source code for the practical part.

Program

1. What is Vue.js?
2. Creating a Vue project with vue-cli
3. Declarative rendering
4. Debugging with Vue.js devtools
5. Conditional instructions and cycles
6. Management of user input
7. Composition of components
8. Data and methods
9. Component lifecycle
10. Template syntax
11. Computed variables and watchers
12. Applying stylesheets
13. Two-way data binding
14. Using plugins
15. Unit testing of components
16. Ajax calls with axios
17. Single-page applications with vue-router
18. Advanced status management with vuex
19. Server-side rendering with Nuxt.js

Inglorious Coderz di Mistretta Matteo Antony
Via Val Della Torre, 39
10149 Torino (TO)

Hangouts: antonymistretta
Skype: antony.mistretta

antony.mistretta@gmail.com
www.ingloriouscoderz.it

Git

> “ I'm an egotistical bastard, so I name all my projects after myself. First Linux, now git. ”
>
> — *Linus Torvalds*

Git is the code versioning tool par excellence: it has superseded the historical CVS and SVN, and has even been promoted by Microsoft at the expense of TFS. Its success is due to its high performance (it is written in pure C), to its distributed nature and to an accurate design that makes it the best friend of any developer.

In this course we describe the philosophy behind Git, the most used commands, and we expose the two most widespread branching strategies: GitFlow and the Cactus Model.

Prerequisites

A basic knowledge of the command line is sufficient.

Duration

The expected duration is 8 hours.

Material

The slides used during the course are available.

Program

1. Git theory
    1. Version control
    2. Local/centralized/distributed versioning
        3. Storing snapshots
        4. Local operations
        5. Integrity checks
        6. Incremental database
        7. The three states
2. Basic commands
    1. init, add, commit, rm, mv
    2. status, diff, log
    3. The .gitignore file
3. Go back
    1. commit --amend
    2. HEAD reset
    3. checkout -f
4. Remote repository
    1. clone, remote
    2. fetch, push, pull
5. Tagging
6. Branching
    1. What is a branch in Git
    2. Move from one branch to another
    3. branch, merge, rebase
    4. Remote branches
7. Distributed workflows
    1. Rules for writing comments
    2. Fork and pull request
8. GitFlow
9. Cactus Model

Inglorious Coderz di Mistretta Matteo Antony
Via Val Della Torre, 39
10149 Torino (TO)

Hangouts: antonymistretta
Skype: antony.mistretta

antony.mistretta@gmail.com
www.ingloriouscoderz.it