



## **Training Schedule**

December 2006

# Training with Instil Software

Welcome to the Instil Training Directory. In this document you will find details on what training is available, and how to register for courses and workshops.

All training is undertaken on-site and in an intensive workshop environment. Candidates are expected to spend the majority of their time working through hands-on exercises, immersed in 'doing' rather than watching slides or listening to a lecture.

To ensure that your specific needs are met, we offer customised versions of all our training courses, as well as post workshop consultancy to help facilitate the introduction of training material into your organisation. Please do not hesitate to contact us for further details on any of these courses or on training requirements not listed below.

## Contact Us

Instil Software Ltd

29 Oakland Avenue,  
Belfast,  
N. Ireland  
BT4 3BW

Telephone: +44 (0)2890 208878  
Mobile: +44 (0)7977 454505  
Email: [info@instilsoft.com](mailto:info@instilsoft.com)

## Booking

Please email: [training@instilsoft.com](mailto:training@instilsoft.com) or call 07977 454505 .

## Pricing

On-site courses: £1000 (+ VAT) per instructor day.

Due to the vocational, hands-on nature of these courses, we recommend that class size is limited to 8 people at most.

### Passport Options

Competitive discounts are available by pre-booking through one of our passport options. Pre-booked training days can be used at any time up to 1 year from booking - training days do not have to be used in 1 slot.

<b>Option</b>	<b>Description</b>	<b>Discount</b>	<b>Rate</b>
<i>Passport 5</i>	5 days pre-booked training per annum	5%	£950
<i>Passport 10</i>	10 days pre-booked training per annum	10%	£900
<i>Passport 20</i>	20 days pre-booked training per annum	20%	£800

## Courses

Passport Options.....	3
Test-Driven Development.....	5
Java Development with the Spring Framework.....	7
Design Patterns Applied.....	9
Java Web Application Development.....	10
.NET Development.....	12

# Test-Driven Development

2-3 days

## Course Description

Test-driven development (TDD) has emerged as possibly the most important productivity enhancing development techniques of recent years. TDD defines an enabling set of practices that equip developers with a methodical and disciplined approach to design, helping simplify an otherwise complex process and ultimately delivering on its promise of 'better code, faster'. This workshop shows you how and why.

This course can be delivered over 2 or 3 days. The 2-day course is ideally suited for advanced programmers already familiar with design and testing principles. The 3-day course follows the same syllabus as the 2-day course, but spends more time on key areas such as the TDD rhythm, and interface discovery through mocking.

### **Testing fundamentals**

- programmer testing versus debugging
- cost and quality
- white and black-box testing
- unit testing
- testing behaviour not methods

### **the xUnit/TestNG family**

- test cases, suites, runners, and fixtures
- asserting expectations
- Ant/NAnt
- Eclipse/Visual Studio

### **refactoring**

- how and when
- removing duplication
- code smells
- intention revealing code
- common refactorings

### **testing single objects**

- the red, green, refactor rhythm
- sufficient design
- faking it
- one to many
- obvious implementation

### **testability and isolation**

- loose coupling and high cohesion
- open closed principle
- dependency inversion
- dependency management

### **testing clusters of objects**

- decoupling techniques; layering, façades, dependency injection
- IoC containers, object wiring and Spring
- interaction versus state based testing
- mocking versus stubs
- static and dynamic mocks
- interface discovery through mocking
- anti-patterns; statics, globals and singletons

### **miscellaneous**

- tdd versus design by contract
- coverage
- continuous integration

### **Audience**

Java or C# developers looking to improve the quality of their code, willing to embrace change and accept new ideas.

### **Pre-requisites**

The majority of attendee time is spent working through hands-on exercises. Attendees should therefore have a solid understanding of Java/C#.

Facilities should include a whiteboard or flip chart; a projector for a laptop; and adequate workstations with your preferred IDE installed.

# Java Development with the Spring Framework

3-5 days

## Course Description

Over the past 2-3 years the Spring Framework has become *the* tool of choice for many large and small-scale Java development projects. The reason? Well, simple really. By providing a simplified, non-intrusive programming model, developers have been freed to focus on delivering core business logic, rather than fighting a technology (EJBs) that caused complexity and pain. Furthermore, with an IoC container at its heart, developers no longer have any left excuses for not testing their own code!

This course can be delivered between 3 to 5 days, depending on your needs and the scope of material delivered. Given the size of the Spring we would not recommend any less than 3 days. The content below is only a small part of what can be delivered.

Note: Instil having been using Spring daily since before V1.0. We provide this course because, not only do we know the technology, we also love using it!

### J2EE

- The problems with J2EE

### IoC Container

- Dependency Injection
- Bean Factory/Application Context
- Constructor/Setter Injection
- XML configuration
- Factory beans
- Bean Post Processors
- Bean Factory Post Processors.

### AOP (Spring 1.2)

- Dynamic AOP
- Pointcuts
- Advice/Advisors/Interceptors
- ProxyFactoryBean
- Autoproxying

### JDBC Abstraction

- JDBC Template
- Transactional and Resource Management

**Remoting**  
**Object/Relational Mapping**  
**Testing with Spring**  
**Designing Applications With Spring**

**Audience**

Java developers wishing to understand how to develop application using Spring, and how to leverage its most important features.

**Pre-requisites**

As with all our courses, the majority of attendee time is spent working through hands-on exercises. Attendees should therefore have a solid understanding of Java and XML. Some understanding of J2EE would certainly help.

Facilities should include a whiteboard or flip chart; a projector for a laptop; and adequate workstations with your preferred IDE installed.

# Design Patterns Applied

3 days

## Course Description

Since its publication 10 years ago, the GoF's seminal book *Design Patterns* has had a profound effect on how we approach object oriented design. This workshop takes a fresh look at these patterns, analysing their strengths and limitations in the context of modern software development, as well as introducing a number of newer patterns and idioms.

### context

- what are patterns
- why do patterns work
- how to use them

### pattern catalog

- Strategy
- Specification
- State
- Decorator
- Template Method
- Immutable Value
- Null Object
- Flyweight
- Façade
- Iterator
- Enumeration Method
- Factory
- Abstract Factory
- Proxy (and Java's dynamic proxy)
- Singleton (as a pattern and an anti-pattern)
- Dependency Injection
- and much more....

## Audience

This course is a mixture of practical exercises, discussion and presentation and is suitable for anyone new to object oriented design or interested in understanding a little more about how to effectively apply design patterns.

## Pre-requisites

A good understanding of Java/C# and, at least a passing familiarity with UML.

Facilities should include a whiteboard or flip chart; a projector for a laptop; and adequate workstations with your preferred IDE installed.

# Java Web Application Development

5 days

## Course Description

Modern web application development can be extremely complex, often involving a mind-numbing mix of languages, frameworks and technologies. This workshop leads you through the development of a complete web application from soup to nuts, imparting the skills to use and combine technology effectively to create robust, maintainable web applications.

### concepts

- dynamic presentation
- frameworks and Model 2 MVC
- layered architecture and separation of concerns

### the web container

- servlets
- filters and listeners
- web.xml deployment descriptor configuration
- building and deploying web applications
- security
- container services

### presentation layer

- markup generation
- event processing
- managing application state
- form handling and validation
- navigation
- JSP/JSTL/JSF/Struts

### application layer

- organising business logic; transaction scripts, service layer
- domain modeling; entities, values, services
- transparent persistence
- testability

### data access layer

- data access and the DAO pattern
- JDBC; connectivity, querying, updating, transactions, patterns.
- object relational mapping; iBatis:
- caching

### miscellaneous

- error handling
- effective use of checked and unchecked exceptions.
- performance and scalability

**Audience**

Developers looking for an understanding on how best to approach, design and develop J2EE web applications.

**Pre-requisites**

A solid understanding of Java and of OO concepts. A decent understanding of HTML, RDBMS concepts and SQL.

Facilities should include a whiteboard or flip chart; a projector for a laptop; and adequate workstations with your preferred IDE installed.

# .NET Development

3-6 days

## Course Description

A highly configurable course that introduces candidates to the fundamental building blocks of the .NET architecture, but at the same allowing you to pick and choose key areas for inclusion. We will work closely with you to tailor this course to your specific needs. For example, this course has been tailored for experienced Java developers moving to .NET for the first time. It has also been taught using VB rather than C#.

### Core Concepts of the .NET Framework

- The evolution of the .NET Framework
- Intermediate Language and the CLR
- The Common Type System (CTS)
- Mixed language development
- Interoperability with J2EE

### Understanding Intermediate Language

- Decompiling code with *ILDASM*
- Creating a simple program in IL
- Compiling the program with *ILASM*
- Understanding and using the call stack
- Manipulating fields and local variables
- Creating objects and calling methods

### Versioning Assemblies

- Strongly naming assemblies
- Publishing libraries to the *GAC*
- Adding code base information
- Redirecting existing bindings

### Automating Builds

- Common issues with build management
- Automating common tasks with *NAnt*
- Automating common tasks with *MSBuild*

### C# Part I

- The structure of a C# program
- Declaring static methods and fields
- Adding and using local variables
- Using *if* and *switch* for conditionals
- Different ways of iterating over code
- Creating single and multidimensional arrays
- Working with strings and *StringBuilder*

## **C# Part II**

- Defining classes and creating objects
- Adding instance methods and fields
- Properties as a special type of method
- Initializing objects using constructors
- The purpose of static constructors
- Creating a hierarchy of classes
- Calling base class constructors
- Overriding methods and properties
- Correctly overriding methods of *Object*

## **C# Part III**

- Working with interfaces and abstract classes
- When and how to use private nested classes
- Costs and benefits of operator overloading
- Using delegates and events for call-backs
- Destructors and using the *Finalize* method
- The *IDisposable* idiom and *using* blocks
- Attributes and aspect oriented development
- When to use a structure rather than a class
- The C# pre-processor and compiler directives
- An overview of how to write unsafe code

## **Collections**

- Sequential versus associative collections
- Creating your own custom data structure
- Making a data structure enumerable
- Arrays, *ArrayList* and the *IList* interface
- Using *Hashtable* and *SortedList*
- Exploring specialized collections

## **Threading**

- Core concepts of concurrent programs
- Concurrency using *Thread* and *ThreadStart*
- Controlling access to shared data using locks
- Advanced locking using the *Monitor* class
- Concurrency using asynchronous delegates
- The IOU Pattern and asynchronous delegates

## **Streams and File I/O**

- Representing paths using *File* and *FileInfo*
- File, network and memory based streams
- Binary streams versus text based streams
- Reading and writing from the file system
- An overview of object serialization in .NET
- Serializing objects via binary formatters
- Serializing objects via XML formatters
- Using streams to encrypting data

## **Networking**

- Review of core TCP/IP concepts
- Basic networking with *TcpClient*
- *WebClient* for HTTP networking
- Understanding the *Socket* class

## **Manipulating XML**

- Review of core XML concepts
- Applying the DOM standard in .NET
- Using *XmlReader* to digest XML files
- Validating against an XML Schema
- Applying XSLT transformations

## **Introducing Inspection**

- Explaining the role of *Type* objects
- Different ways of finding *Type* objects
- Examining the members of a type
- Examining attributes attached to types

## **Advanced Introspection**

- Creating objects and calling methods
- Setting fields and properties in objects
- Loading new assemblies into programs
- Dynamically generating assemblies

## **The Code DOM library**

- Introduction to code generation
- Overview of abstract syntax trees
- Creating a skeleton syntax tree
- Adding methods and fields
- Adding statements to methods
- Compiling code using providers
- Generating code using providers

## **Introducing Windows Forms**

- Manually creating a WinForms GUI
- Creating a GUI in Visual Studio
- Adding event handlers for controls
- Binding controls to data sources
- Creating your own controls

## **Fundamentals of ADO.NET**

- Key principles and features of ADO .NET
- Providers included with the framework
- Opening and managing connection objects
- Running simple queries using commands
- Using a *DataReader* to pipeline query results
- Running multiple queries through a reader

### **Working with Data Sets**

- The internal structure of a *DataSet* object
- Using an adapter to populate a *DataSet*
- Storing multiple tables in a *DataSet*
- Populating *DataSet* objects from XML
- Creating and using typed *DataSet* classes
- Configuring a *DataSet* to be updatable

### **Introduction to ASP.NET**

- Comparing ASP, JSP and ASP .NET
- The process of page compilation in detail
- Code behind files verses partial classes
- How a request is processed in the runtime
- Events fired during the lifecycle of a page
- ASP .NET directives and tracing events
- Writing pages using scripting elements
- Storing state in the session and application
- Dispatching requests between pages

### **Working with Controls**

- The concept of a tree of controls
- Adding HTML controls to a page
- The lifecycle of an ASP .NET control
- How controls use the *\_VIEWSTATE*
- Attaching handlers to control events
- Framework level events

### **Advanced Controls**

- Adding Web Controls to pages
- Binding controls to data sources
- Using templates to display data
- The *DataGrid* control in detail
- Creating a custom ASP .NET control
- Adding data binding to the control
- Adding state management

### **Core Concepts of Web Services**

- Creating and understanding XML Schemas
- Reading and customizing WSDL documents
- Document and RPC based SOAP messages
- Using UDDI to publish and search for services
- The evolution of Service Oriented Architectures

### **Creating and Deploying Web Services**

- Building an ASP .NET Web Service
- Exposing simple methods as services
- Returning objects as XML documents
- Customizing the structure of the XML
- Using the Web Service from Java
- XML Signature and Encryption

**Audience**

This course is a mixture of presentation, demonstration and plenty of hands-on exercises, and is suitable for anyone new to the .NET framework.

**Pre-requisites**

A good understanding of programming basics and, ideally, another high-level objected oriented programming language like Java.

Facilities should include a whiteboard or flip chart; a projector for a laptop; and adequate workstations with your preferred IDE installed.