Seminars
ON DEMAND

QA ACADEMY
Skills for better Software

Know-How –
Fuelling better results

Seminars

for better software

QA SYSTEMS
The Software Quality Company

Driving embedded
software quality

Testen von Software eingebetteter Systeme

Teilnehmerstimmen:

„Mir hat der Kurs sehr gut gefallen und werde ihn auch meinen Kollegen weiterempfehlen. Die vorgestellten industrie-relevanten Beispiele fand ich sehr hilfreich, um die verschiedenen Konzepte zu verstehen, und außerdem der systematische Aufbau des Kurses hat dem besseren Verständnis auch beigesteuert.“

„Ich habe nichts gefunden, was die Beurteilung schlechter als :-) :-) erscheinen lässt.“

“Sehr guter Praxisbezug. Ich hätte das Seminar bereits in einem früheren Projektstadium brauchen können. Auch für Projektleiter und Manager zu empfehlen! Sehr kompetenter Referent.”

Ihr Nutzen

Lernen Sie mit Stephan Grünfelder, wie das Testen von Software effizient, effektiv und keinesfalls langweilig wird. Wie entwirft man Tests? Worauf kommt es beim Kauf von Testtools an? Wie bewertet und verbessert man Testprozesse? An diesen zwei Tagen erhalten Sie die Antworten und einen umfassenden Überblick über Testmethoden und Testmanagement im gesamten Softwarelebenszyklus. Stephan Grünfelder stützt sich auf langjährige Erfahrung als Softwaretester und Projektleiter im Automotive-Umfeld, der Medizintechnik und der unbemannten Raumfahrt. Er ist Autor zahlreicher Publikationen in der deutschen Fachpresse, Hochschullektor, und ist nach wie vor selbst aktiv in der Softwareentwicklung tätig. Seine Seminare vermitteln praxisnahes Wissen auf höchstem Niveau in leicht erlernbarer Form.

Zielgruppe

- Softwareentwickler
- Tester
- Technische Projektleiter mit/ohne Vorkenntnisse

Voraussetzungen

Keine. Grundlagenkenntnisse der Programmiersprache C erleichtern das Verstehen einiger Fallbeispiele.

Unterlagen

Neben den Vortragsunterlagen erhalten Teilnehmer ein Nachschlagwerk in Buchform.

Überblick

Der Kurs gibt einen umfassenden Überblick über Testmethoden und Testmanagement im gesamten Softwarelebenszyklus. Sie bekommen das nötige Wissen, um selbst Tests zu entwerfen und Entscheidungen beim Kauf von Testtools treffen zu können. Sie werden befähigt Testprozesse in Ihrer Firma zu bewerten, zu verbessern und testrelevante Kapitel von Standards wie IEC 61508 und DO-178B zu verstehen.

Jeder Abschnitt des Kurses ist mit Beispielen aus der industriellen Praxis illustriert. Das Beherrschen des Testhandwerks allein ist aber zu wenig, um die Softwarefehlerrate zu reduzieren. Aus diesem Grund zeigt dieser Kurs auch wie Sie Testdurchführung und Testmanagement mit anderen Methoden verzahnen müssen, um das Kostenoptimum anzusteuern.

Testen von Software eingebetteter Systeme

Programm

1. Tag

Nachdem häufige Ursachen von Bugs untersucht wurden, werden verschiedene Verifikationsmethoden und ihr Platz im Softwareentwicklungszyklus gezeigt. Die Teilnehmer lernen, wann man welche Testschritte in welchem Detaillierungsgrad plant und wann man diese dann am besten ausführt. Dem Wechselspiel von Softwareanforderungen und Test wird dabei besonderes Augenmerk gewidmet, weil Missverständnisse bei der Kommunikation mit dem Kunden oft die teuerste Art von Fehlern darstellen. Daher sind auch Reviews von Anforderungen Teil des Kursinhalts.

Code-Review, Unit-Test, Integrationstest und Softwaresystemtest im Zusammenspiel. Die Teilnehmer erkennen, welche Art von Fehlern in welcher Teststufe gefunden werden kann. Zu jeder Teststufe werden Grade von Testabdeckungen definiert und miteinander bezüglich Aufwand und Schärfe verglichen.

White Box und Black Box Testing wird genauer vorgestellt, dabei wird auch darauf Bedacht genommen, wie man Testaufwand schätzen kann und Testgüte beurteilen. Für beide Techniken werden Entscheidungshilfen gegeben, die helfen zu beurteilen, ob sich Testautomatisierung lohnt und welche Art von Tools dazu benötigt werden. Screenshots und Live-Demonstrationen von Tools geben Orientierungshilfen zu den Themen: Stress Test, Performance-Test, Load Test, Volume Test, Installation Test, Ressource Test, Security Test und Regression Test.

Als Abschluss des ersten Kurstages können die Teilnehmer ihre neuen Fähigkeiten messen. Sie entwerfen einen Unit-Testfall und einen Softwaresystemtestfall und bekommen dazu qualifiziertes Feedback.

2. Tag

Heute werden Methoden der in der Literatur wenig behandelten Integrationstests vorgestellt. Die Teilnehmer lernen, wie man diese Tests systematisieren kann und wann darauf verzichtet werden kann. Die Verbesserung der Qualität von Code-Reviews wird Thema sein und ergänzend dazu werden Software-Coding-Standards und Stärken und Schwächen von statischen Code-Analyse-Tools vorgestellt. Einsatz von Testprozessmetriken zur Steuerung der Testeffizienz und als Instrument des Managements. Race Conditions können zu einer ganz besonders unangenehmen Art von Fehlern führen und werden mit traditionellen Testmethoden nicht oder nur durch Zufall gefunden. Welche Tools können automatisch solche Fehler erkennen? Stärken-Schwächen-Vergleich von Tools.

Einblick in neue Entwicklungen am Sektor Softwaretest: modellbasiertes Testen, automatische Testfallerstellung, Worst Case Execution Timing Analysis. Rechtliche Aspekte und wichtige den Test ergänzende Methoden. Analyse von Managementfehlern bei gescheiterten Testprojekten.

Über den Referent Dr. Stephan Grünfelder

Stephan Grünfelder studierte Informatik an der TU Wien und promovierte ebendort im Gebiet der Robotik. Seit dem Studium blieb er eingebetteten Systemen stets treu: er entwickelte Testsoftware für Telekommunikations-Satelliten, Firmware und Testsysteme für EEGs, war vier Jahre als Projektleiter für die Entwicklung von Motorsteuergeräten tätig und leitet seit 2010 das Testautomations-Team eines Unternehmens für Broadcasting-Elektronik. Stephan Grünfelder ist Lektor an der FH Technikum Wien, Autor des Buchs “Software-Test für Embedded Systems” und wurde für Beiträge zu den Themen Test und Management vom WEKA-Zeitschriftenverlag zwei Mal mit dem Preis für den besten Fachartikel des Jahres ausgezeichnet.

C++1x for Embedded Systems compact

Testimonials:

”Understandable explanation & collegial atmosphere :-)“

“Andreas Fertig is very competent and conveys the topic in an understandable and entertaining way. Merci!”

”Very good seminar, where illustrative examples are shown.”

Benefits

The flexibility of the C ++ language has improved even further with the new features of the C++11 standard and its successors. Some of these innovations are particularly interesting for embedded systems. For example it is now possible to move computations from run to compile time, resulting in a smaller and more performant codebase.

Are you also tired of writing ++i and still trying to get > and >= right? C++1x helps by leaving recurrent code of this type to the compiler. As a result, you gain more time to write really sophisticated code.

In this four-day seminar Andreas Fertig teaches how C++1x supports you to write faster and more robust code. After the seminar, you will write clearer and more precise code and will pass code reviews easier and more confidently, among other things.

Target group / Prerequisites

- Developers and architects of C++ based systems. Familiarity with the major features of C++ is required. Concepts and technical innovations of C++1x will be introduced.

Format

Interactive class with exercises. All attendees are encouraged to play with the course material during the class.

Overview

In this seminar, attendees C++1x familiarise themselves with the use of embedded systems. Although C++1x is not specifically designed for embedded systems, the language contains various valuable features for such systems. Participants gain knowledge to optimise compiled code size and performance. An example is static_assert. With this function, it is possible to check conditions at compile time. Calculations can also be done with constexpr functions at compile time. In both cases, these features positively affect runtime behaviour and code size.

The use of move semantics promises faster and more efficient copying operations. Participants will learn how move semantics work, how to use it, and in what situations the compiler will disable them.

During the seminar, the participants will learn which new features of C++1x they can use in their codebase. Complex problems can be formulated more easily using the extended support of the compiler. The resulting code is often more robust and less error-prone. Participants who work with an older compiler, without C++1x, will receive tips and suggestions on using new features in their code. After the seminar, participants write clearer and more precise code. For example, participants will pass code reviews easier and safer.

C++1x for Embedded Systems compact

Topics Outline

- Clean and modern C++ code
- Do things at compile time: constexpr, constexpr if
- The right choice at the right time: preprocessor, inline or constexpr
- ROMability
- Interesting C++1x elements for embedded systems: override, auto, literaloperator, nullptr, structured bindings, std::optional
- A must: range basd for loops
- Influence of C++1x to performance and code size
- Templates: the proper dose
- Usage of the new initializerlist
- Lambdas
- Move semantics

About the speaker Andreas Fertig

Andreas Fertig, CEO of Unique Code GmbH, is an experienced trainer and lecturer for C++ for standards 11 to 20. Andreas is involved in the C++ standardisation committee, in which the new standards are developed. At international conferences, he presents how code can be written better. He publishes specialist articles, e.g., for iX magazine, and has published several textbooks on C++. With C++ Insights (https://cppinsights.io), Andreas has created an internationally recognised tool that enables users to look behind the scenes of C++ and thus to understand constructs even better. Before working as a trainer and consultant, he worked for Philips Medizin Systeme GmbH for ten years as a C++ software developer and architect focusing on embedded systems.

Software Inspections

Testimonials:

“Everything was presented in an extraordinarily understandable and exciting way. My expectations were more than met.
It's amazing what you can still learn here about reviews.“

“Many thanks for new ideas, which I will try to implement in my company. Especially the practical part was varied and fun. Keep it up!”

“Highly recommended! Every SW engineer should be made to attend!!!"

Benefits

Reviews and inspections can be conducted independent of process model, development environment, and programming language; they are therefore a useful tool for the entire software industry. Join Peter Rösler for two days and learn how to plan and lead a review as a moderator, and how to evaluate the results quantitatively and qualitatively.

Target group

- Software developers
- Project managers
- QA representatives
- Software testers

Handouts

In addition to the seminar materials, each participant receives the book ”Reviews in der System- und Softwareentwicklung” by Rösler / Schlich / Kneuper (German language) or, if preferred, “Software Inspection” by Gilb/Graham (English language).

Furthermore, the seminar price includes fifteen hours of online support by the trainer. These can be used by the participants as and when required, for example, if they are to lead their first review as a moderator after the seminar.

Overview

Reviews improve not only SW quality – they also increase the productivity of SW projects, a consideration frequently overlooked. Productivity can be increased by as much as 25 - 35 % if reviews are conducted properly, i.e. if the critical success factors of reviews are taken into account.
Reviews such as these are also referred to as "formal inspections", "Fagan/Gilb-style inspections" and "peer reviews". Since 1975, many organisations have carried them out successfully. The main aim of these reviews is to identify errors in text documents or programs at an early stage. Such reviews are to be distinguished from walk-throughs / presentation reviews, in which the author presents the content of a document, or management reviews / project status reviews, in which decisions are made as to whether and how something should be carried out.

Software Inspections

Program

Day 1

- Introduction to reviews and inspections.
- Types of review.
- The phases of a review.
- Participant roles.
- 15 critical success factors.
- Agile software development and agile inspections.
- Video scenes demonstrating appropriate and inappropriate conduct of reviews.

Day 2

- Exercise: conducting a real review
  For this exercise, participants will check a C-program against its specifications, using checklists. (The course material also contains an exercise for reviewing text documents. If preferred, the participants can review the text document instead.)
- Tools.
- Work aids and forms.

About the speaker Dipl.-Inform. Peter Rösler

Peter Roesler was born in 1961 and studied computer science at the Technical University of Munich from 1981 to 1987. He has worked as a programmer, chief designer, QA manager and project manager for a number of software development projects, predominantly in the airports/airlines division of Softlab in Munich. He is a trained inspection leader for Gilb inspections and has already given over 190 inspection seminars to more than 1,800 engineers. He has been working as an independent inspection trainer and consultant from 2005 to 2016 and is Quality Manager Software Development at Carl Zeiss Microscopy GmbH since 2016.

MISRA C:2023– The rules for the development of safety-critical Software

Testimonials:

"Super seminar, perfectly organised, expectations were fully met."

"The examples were very interesting and helped me to understand many things better. Seminar is recommendable."

“Dry topic brought across well.”

Benefits

In this one-day seminar learn about the objective of MISRA rules for error prevention in safety-critical software. In addition to the practical applicability of the rules, the integration of rule checking in the development process will be discussed. How to achieve MISRA conformance?

Target group

- Software developer
- Technical project manager

Prerequisites

- Any C Compiler that is compatible with ANSI C89 or ISO C90 can be used. For MISRA C:2023 the compiler may also support C99 or the C11 standard, which is the case for most compilers in use today. The recommended platforms are Windows, MacOSX, UNIX or Linux systems.
- Course participants should master the C programming language and simple data operations on the system they use. A selection of rules are explained with real program examples..

Handouts

No training documents will be handed out by QA Systems or the speaker. The training is based on the MISRA C:2023 rule set. It is important that each participant has access to the document during the seminar. The guidelines can be purchased by the clients themselves via https://misra.org.uk/product/misra-c2023/.

Overview

In this course, the development of safety-critical software written in ANSI/ISO C in the variants C89/90, C99 and C11 for embedded systems in the automotive field is covered. The possible reasons for failure are analysed and discussed. The typical failures occurring in the implementation with standard C are classified and associated to the MISRA rules. Beside the work on the rules, the structure of the MISRA Rule Document from 2023 is discussed. The rules themselves are covered in detail and the documentation requirements necessary to prove the compliance with the rules are discussed.

MISRA C:2023 – The rules for the development of safety-critical Software

Topics Outline

- The Motivation behind the MISRA rule catalogue.
- The Error in Software.
- The Error in Coding (Implementation).
- Thoughts on Development of Embedded Systems.
- Safety-relevant and safety-critical Software.
- Special Sources of Error in C.
- Undefined Behaviour of C.
- Implementation-dependent Behaviour of C.
- Common Errors in C.
- Legibility and Clearness of C Code.
- The C Language and the Developer's Intuition.
- Improved Description of the Rules compared to older MISRA Standards.
- Simplification of the Rule Description.
- Clear Classification of Rules and Guidelines.
- Automatic Verifiability.
- Rule Scope.
- Required and Recommended Rules.
- Rules for ISO C99.
- Rules for ISO C11
- Static Code Analysis.
- Programming Guidelines and Coding Styles.
- The MISRA Compliance Document
- Development Process.

About the speaker Ralf Schneeweiß

In several projects, before and during his freelance work, Ralf Schneeweiß acquired profound knowledge in the practical use of C and C++ on several operating systems. In addition to the practical knowledge, he has also mastered the theoretical basics of these programming languages. In 1989 he learned C and developed in the university area. With C++, Mr. Schneeweiß came into contact with object-oriented programming for the first time in 1990 and has followed the development of this technique ever since. The first contact with Java took place in 1997 and at the same time the internalisation of pattern-oriented software development.

Since 1990 his professional career took place across many IT projects besides his studies. He first delivered professional training in 1995 for C, C++, some older programming languages, application programs and Lotus Notes. Mr. Schneeweiß has undertaken many freelance projects since 1995, becoming a full-time freelance consultant in 1998.

MISRA C++:2023 – The rules for the development of safety-critical Software with C++

Benefits

The new MISRA C++:2023 standard was published in October 2023. The new 2023 standard not only replaces the old 2008 standard but **also supersedes the Autosar C++14 standard**, which has contributed to its development.

This two-day seminar will assist you in applying and thoroughly understanding the standard.

Target group

- Software developer
- Technical project manager

Prerequisites

- Any ISO-compatible C++ compiler that supports C++17, the ISO/IEC 14882:2017 standard, can be used. This is the case with most current compilers today. The recommended platforms are Windows, MacOSX, UNIX, Linux or QNX systems.
- Course participants should master the C++ programming language and simple data operations on the system they use. Some rules are run through with real program examples.

Handouts

No training documents will be handed out by QA Systems or the speaker. The training is based on the MISRA C++:2023 rule set. It is important that each participant has access to the document during the seminar. The guidelines can be purchased by the clients themselves via https://misra.org.uk/product/misra-cpp2023/.

Overview

In this MISRA C++ course, the development of safety-critical software with ISO-C++17 in embedded systems in the automotive field is examined. For this purpose, the failure possibilities in a software project are analysed and discussed.

The typical failures occurring in the implementation with ISO-C++17 are classified and associated to the MISRA rules. For this purpose, the structure of the composition of the MISRA rule document from 2023 is reviewed. The rules themselves are covered in detail. In addition, the documentation and process requests necessary to reaching the compliance with the rules are debated.

The objective of the seminar is to get to know the principles and rules that MISRA offers for failure prevention in C++ projects. Additionally, the problem areas touched by the MISRA rule catalogue in C++17 should be thoroughly understood.

Topics Outline

- The Error in Software
- The Error in Implementation and its Causes.
- Thoughts on Development of Embedded Systems
- Safety-relevant and Safety-critical Software
- Typical Sources of Error in C++
- Special Sources of Error in C++
- Undefined Behaviour of C++
- Implementation-dependent Behaviour of C++
- Legibility and Clearness of C++ Code
- The Safe Use of Object-oriented Concepts in C++
- The MISRA-C++ Rule Catalogue
- Required and Recommended Rules
- Static Code Analysis
- Programming Guidelines and Coding Styles
- Development Process.

About the speaker Ralf Schneeweiß

In several projects, before and during his freelance work, Ralf Schneeweiß acquired profound knowledge in the practical use of C and C++ on several operating systems. In addition to the practical knowledge, he has also mastered the theoretical basics of these programming languages. In 1989 he learned C and developed in the university area. With C++, Mr. Schneeweiß came into contact with object-oriented programming for the first time in 1990 and has followed the development of this technique ever since. The first contact with Java took place in 1997 and at the same time the internalisation of pattern-oriented software development.

Since 1990 his professional career took place across many IT projects besides his studies. He first delivered professional training in 1995 for C, C++, some older programming languages, application programs and Lotus Notes. Mr. Schneeweiß has undertaken many freelance projects since 1995, becoming a full-time freelance consultant in 1998.

Autosar – The rules for developing critical software with ISO C++14

Testimonials:

"A great wealth of experience, very good explanations."

"Understandable and helpful examples! Very competent speaker!"

"Excellent, expectations fully met!"

Benefits

In this one-day seminar you get to know the principles and rules that Autosar C++14 offers for failure prevention in C++ projects.

Target group

- Software developer
- Technical project manager

Prerequisites

- Any C++14 compatible compiler can be adopted for this purpose. The recommended platforms are Windows, MacOSX, UNIX or Linux systems.
- The only important thing is that course participants master the C++ programming language and simple data operations on the system they use. A few rules are run through with real program examples.

Handouts

No training documents will be handed out by QA Systems or the speaker. The training is based on the Autosar C++14 Guidelines AND MISRA C++:2023 rule set. It is important that each participant has access to both documents during the seminar. The MISRA C++ guidelines can be purchased by the clients themselves via https://misra.org.uk/product/misra-cpp2023/.

Overview

This seminar handles the development of critical software with ISO C++14 in embedded systems within the automotive field. For this purpose, the failure possibilities in a software project are analysed and discussed with C++14. The basic rules from the rule catalogue are also examined. Backgrounds are explained and special features of the C++ language discussed. In addition to theory, practical examples are performed in order to understand the meaning of the rules.

Autosar – The rules for developing critical software with ISO C++14

Topics Outline

- The Error in Software
- The Error in Coding (Implementation)
- Thoughts on Development of Embedded Systems
- Safety-relevant and Safety-critical Software
- Typical Sources of Error in C++
- Special Sources of Error in C++
- Undefined Behaviour of C++
- Implementation-dependent Behaviour of C++
- Legibility and Clearness of C++ Code
- The Safe Use of Object-oriented Concepts in C++
- The Autosar C++14 Rule Catalogue
- Required and Recommended Rules
- Static Code Analysis
- Programming Guidelines and Coding Styles
- Development Process.

About the speaker Ralf Schneeweiß

In several projects, before and during his freelance work, Ralf Schneeweiß acquired profound knowledge in the practical use of C and C++ on several operating systems. In addition to the practical knowledge, he has also mastered the theoretical basics of these programming languages. In 1989 he learned C and developed in the university area. With C++, Mr. Schneeweiß came into contact with object-oriented programming for the first time in 1990 and has followed the development of this technique ever since. The first contact with Java took place in 1997 and at the same time the internalisation of pattern-oriented software development.

Since 1990 his professional career took place across many IT projects besides his studies. He first delivered professional training in 1995 for C, C++, some older programming languages, application programs and Lotus Notes. Mr. Schneeweiß has undertaken many freelance projects since 1995, becoming a full-time freelance consultant in 1998.

All seminars can be hold as inhouse format, online and onsite.
Our specialists are happy to adapt the content, on-request, to meet your needs.

To find out more, please contact us via email to
seminare@qa-systems.de

QA ACADEMY
Skills for better Software