Rumos valorizamos profissionais!

Certified Scrum Master & Developer (CSM&D)

Percurso

- **Formato do curso:** Presencial
- **Localidade:** Lisboa
- **Data:** 05 Fev. 2019 a 04 Abr. 2019
- **Preço:** 2600€
- **Horário:** Laboral - das 09h00 às 17h00
- **Duração:** 40 horas

SCRUM é uma framework que, quando usada individualmente ou em conjunto com outros processos promove uma maior flexibilidade, visibilidade e produtividade para a empresa.

Este percurso permite-lhe frequentar e adquirir as competências e respetivas certificações Certified Scrum Master e Certified Scrum Developer.

No curso **Certified Scrum Master (CSM)**, os participantes irão aprender tudo o que é necessário para começarem a utilizar o SCRUM.

O curso **Certified Scrum Developer (CSD)** destina-se a software developers (programadores) que desenvolvem software em ambiente Scrum.

O objectivo é ensinar aos formandos as ferramentas e técnicas mais importantes que devem ser utilizadas na construção de um bom software da forma iterativa e incremental que o Scrum proporciona. Estes conceitos são fundamentais a todo o desenvolvimento de software Agile.

O curso inclui:

- Exame de certificação CSM;
- Inscrição na Scrum Alliance;
- Atribui 40 PDUs do PMI: CSM: 16 PDUs + CSD: 24 PDUs

Formador

Chet Hendrickson

Has been involved with Agile Software Development since 1996, when as a member of Chrysler’s C3 project he helped develop Extreme Programming. In 2000, Ron Jeffries, Ann Anderson, and Chet wrote Extreme Programming Installed. It detailed XP’s core practices, how to do them, and how they work together to help teams be successful.

Since 2002, Chet has been an independent consultant, coach, and trainer. In 2009, he was asked by the Scrum Alliance to help develop the Certified Scrum Developer program. Chet and Ron Jeffries taught the first CSD course and continue to offer them in the United States and Europe. He has been a Certified Scrum Trainer since 2009.

Chet and Ron Jeffries often work together and are popular conference speakers, bringing an interesting mix of humor and deep knowledge, and the odd cat picture.

Destinatários

The course is appropriate for individual software developers and testers, as well as development team members who want to experience Agile development for the first time, or to sharpen their skills in Agile.

- Software Development Managers
- Software Development Architects
- Software Developers
- Software Coders
- Software Testers
- Team Leaders.

Pré-requisitos

The candidates for this course should generally be programmers in a programming language such as Java, C#, Swift, C++, Ruby, and so on.

The course involves actual programming in a team environment. Some courses offer student slots to testers, who should have some experience with test automation. Some also accommodate Product Owners, who will help direct the iterations that the programmers and testers experience during the course. This notwithstanding, the course is a programming course, not a survey, overview, or discussion. Actual code will be written, tested, and reviewed by the instructors.

Students should also arrive at the course familiar with:

- Scrum Guide;
- Agile Manifesto 4 values and 12 principles;
- The Scrum Values;

To achive the certification **Certified Scrum Developer (CSD)**, the student must already be a Certified Scrum Master (CSM).

Objectivos

By the completion of the course, the learner will be able to perform test-driven development, acceptance test-driven development, refactoring, and continuous integration in a course-provided sandbox situation. The learner will be familiar with the terms and practices involved, and with why these practices are important. They will have taken the first steps toward becoming expert in Scrum-style iterative and incremental development.

By the completion of the course, the learner will be able to . . .

- Agile Values
    - Define simplicity, communication, and feedback (in relation to the Agile Values that drive Scrum).
    - Describe “individuals and interactions over process and tools.”
    - Describe “working software over comprehensive documentation.”
    - Describe “customer collaboration over contract negotiation.”
    - Explain, using examples, “responding to change over following a plan.”
- Scrum – Study of Scrum principles and practices, including but not limited to, the following key concepts:
    - Define Scrum roles, activities, and artifacts.
    - Outline the process of working with a product backlog and a sprint backlog.
    - Define a sprint.
    - Describe the process of defining “Done.”
- Architecture and Design – Study of architecture and design, focusing primarily on the principles that better enable testability and ease of refactoring, including but not limited to, the following key concepts:
    - Outline at least three principles of architecture in an Agile environment.
    - Design at least one practice on an Agile team.
    - Outline at least two principles that enable testability and ease of refactoring.
- Collaboration – An in-depth look at the way Agile teams work together. This might include, but is not limited to, the following concepts:
    - Describe “working together as one team.”
    - Describe how to “include the customer” in the process.
    - Define pair programming.
- Test-Driven Development – A study of test-first development, including but not limited to, the following concepts:
    - Describe Test-Driven Development (TDD) as a design approach.
    - Review the steps of the red-green-refactor cycle.
    - Explain, using examples, at least three unit testing principles and practices.
    - Outline five qualities of a good test.
    - Describe how to measure test effectiveness.
- Refactoring – An introduction to the practice of refactoring, including but not limited to, the following concepts:
    - Describe when to refactor.
    - Outline refactoring for maintainability.
    - Define refactoring to patterns.
- Continuous Integration – An introduction to the key practices of continuous integration, including but not limited to, the following key concepts:
    - Define a single command build.

    - Summarize how to create a build that is automated, self-testing,and fast.
    - Describe the importance of a single-source repository.
    - Define increasing visibility and automating deployment.

Programa

Certified Scrum Master

- Overview of Scrum
- Sprints
- The ScrumMaster
- The Product Owner
- Product backlog
- Meetings
- Sprint planning
- Release planning
- Tracking progress
- The team
- Scalability

Certified Scrum Developer

The first day begins with a demonstration of pair-programming and test-driven development and an overview of Fitnesse. After this, we move on to the laboratory project that will be the basis for the rest of the course.

After an introduction to the problem and its associated backlog, each team will conduct a Sprint Planning meeting and begin work. At the conclusion of each ninety minute Sprint, each team will review their completed features with the rest of the class and then conduct a Sprint Retrospective.

During each Sprint Review, the instructor will review each team’s application and test code. He will focus on two areas, the relationship between a module and its tests, and the design and structure of the code. It has been understood for over forty years that high quality software is composed of loosely coupled modules of highly cohesive code. These attributes are, to the skilled observer, obvious in the source code and will be the basis of most of the discussion during the code review.

The course proceeds with planning, sprint execution, review, and retrospective (and a short lecture here and there) until the afternoon of the last day. The course concludes with a course review, retrospective, and plenty of time for questions and answers.