iscte
INSTITUTO UNIVERSITÁRIO DE LISBOA

emprego digital

Módulo 3: Princípios de Desenvolvimento de Software

Aula 1
Introdução à Engenharia de Software

UPskill

Sobre o módulo

O módulo de Princípios de Desenvolvimento de Software pretende promover a compreensão e aquisição de competências iniciais e fundamentais da engenharia de software. Os conteúdos programáticos vão passar pelas boas práticas de desenvolvimento de software, o processo de engenharia de requisitos, padrões e metodologias de análise e desenvolvimento de software e metodologias de testes.

Programa do módulo

Aula	Tema
1	Introdução à Engenharia de Software; Git; GitHub
2	Metodologias de Desenvolvimento; Waterfall; Agile; Tools; Requisitos
3	Use Cases; **Introdução ao Projeto**
4	Diagrama de Estados; Projecto
5	Diagrama de Atividades
6	Diagrama de Sequências
7	Projeto
8	Projeto
9	Projeto; Sprint
10	Testes; White-box; Black-box; Teste unitários, integração, carga e usabilidade
11	Análise de Código (Linting); Projeto
12	**Teste (3,5h) - Teste de Engenharia de Software**
13	Projeto
14	Projecto; Final Sprint; Projecto - Início de Implementação

iscte INSTITUTO UNIVERSITÁRIO DE LISBOA

emprego digital

Avaliação do módulo

Teste - Engenharia de Software Classificado de A-E	Aula 12 26 / 01 / 2021	**50%**
Projeto Classificado de A-E	Aula 14 28 / 01 / 2021	**50%**

iscte INSTITUTO UNIVERSITÁRIO DE LISBOA emprego digital

O que é a Engenharia de Software?

Engenharia de software é um ramo da ciência da computação que inclui o desenvolvimento e construção de sistemas e software. Estes podem ser programas que incluem ferramentas de computação e sistemas operacionais. Na sua essência a Engenharia de Software é a aplicação dos campos de engenharia no ramo do desenvolvimento de software.

Campos da Engenharia de Software

- **Requisitos** de Software
- **Desenho** de Software
- **Desenvolvimento** de Software
- **Testes** de Software
- **Manutenção** de Software

Requisitos de Software

Processo de **definir**, **documentar** e **manter** requisitos no processo de desenho de engenharia de software.

Foco nas tarefas que determinam as necessidades dos clientes ou as condições necessárias para o desenvolvimento, alteração ou continuidade de um projeto ou produto.

Permite ter uma lista de tarefas em mão e contacto entre os desenvolvedores e gestores.

Tipos de Requisitos

- Requisitos do **Cliente**
- Requisitos de **Arquitetura**
- Requisitos **Estruturais**
- Requisitos de **Comportamento**
- Requisitos **Funcionais**
- Requisitos de ***Performance***
- Requisitos de **Design**

Requisitos do Cliente

Requisitos operacionais que podem ser definidos ao responder às seguintes perguntas:

- Onde é que o sistema vai ser usado?
- Como é que o sistema vai realizar o objetivo especificado?
- Quais são os parâmetros críticos a ter em conta?
- Que componentes se podem subdividir do objetivo final?
- Quão eficiente ou eficaz é que precisa ser o sistema?
- Durante quanto tempo vai ser usado o sistema?
- Em que ambientes é expectado funcionar?

Desenho de Software

O Desenho de Software pode ser definido por todas as atividades envolvidas na conceptualização, enquadramento e homologação do sistema ou projeto visionado.

Passa pelo planeamento e resolução de problemas na especificação **antes** do desenvolvimento.

Desenvolvimento de Software

A atividade principal na construção do software, a combinação da programação com os vários processos de definição, implementação, verificação, gestão e melhoria do ciclo de desenvolvimento.

Testes e Manutenção

Dois processos que interagem mutuamente no processo da engenharia e desenvolvimento dos projetos, que envolve a investigação técnica que permite averiguar a qualidade do produto ou serviço e que definem as atividades necessárias para a continuidade e coerência do software.

Paradigmas, Modelos e Metodologias

Dentro da engenharia de software encontramos práticas comuns na descoberta e desenvolvimento de soluções que já foram especificadas e usadas em ambientes de produção reais:

- **Agile**; Cleanroom; Incremental: **Prototyping**; Spiral; V model; **Waterfall**
- **Scrum**; Kanban; **Sprint**; **DevOps**

Controlo de Versionamento

Sistemas que permitem manter um histórico de alterações a ficheiros e pastas de um projeto. Desenhadas para coordenar trabalho entre equipas de desenvolvimento de software para trabalharem no mesmo código ao mesmo tempo.

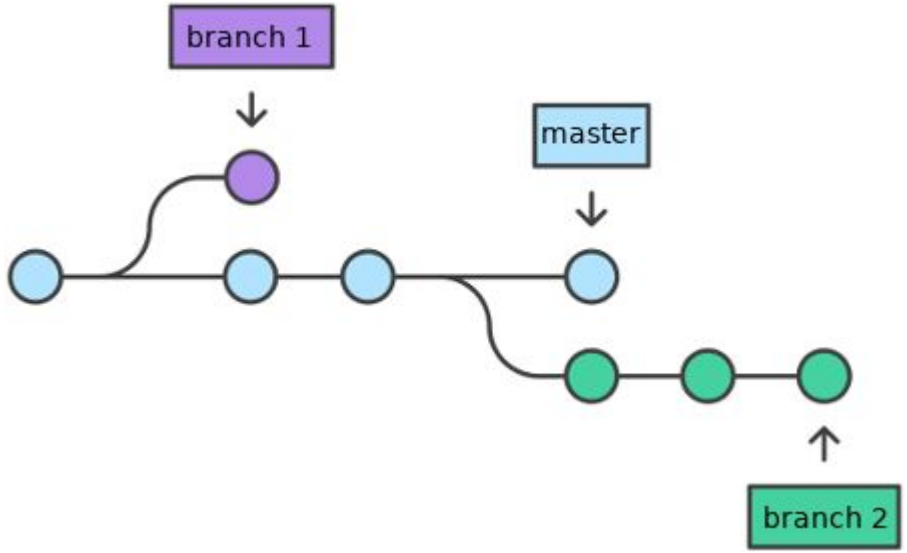
- Subversion (SVN), Mercurial, **Git**, CVS, etc.

Git

O Git distingue-se dos outros sistemas de controlo de versionamento ao permitir acesso completo a todos os ficheiros, *branches* e iterações de um projeto. Permite aceder ao histórico completo de todas as mudanças que foram efetuadas desde o ínicio do repositório e ter acesso a um log transparente de quem e quando se fez as alterações.

Git - Funcionamento Geral

O Git permite criar vários ramos (***branches***) de desenvolvimento que por sua vez permitem distinguir funcionalidades a serem feitas, versões ou ambientes *(dev, QA, staging, prod*).
Num ambiente estável de desenvolvimento, cada alteração concreta no projeto é "gravada" num branch através de ***commits.***



Git - Comandos Básicos

- `git init` - Inicializa um novo repositório git no diretório atual
- `git clone` - Cria uma cópia local de um projeto já existente remotamente
- `git add` - Marca um ficheiro ou diretório como “modificado”
- `git commit` - Grava as modificações no histórico do projeto
- `git push` - Atualiza o repositório remoto com commits feitos localmente
- `git status` - Mostra as modificações pendentes ou à espera de commit

Git - Gestão de *Branches*

- `git branch` - Mostra os *branches* locais
- `git checkout` - Altera o *branch* atual para o especificado
- `git merge` - Junta dois *branches*
- `git revert` - Retrocede commits existentes e as alterações introduzidas
- `git cherry-pick` - Aplica alterações de um commit de outro branch ao atual

Git - Exercício 1

Objetivo: Criar e inicializar um novo repositório chamado “hello-world” no GitHub, adicionar um ficheiro *README* ao projeto, marcar a alteração no ficheiro criado, fazer um commit com uma mensagem adequada, adicionar o url do github como remote ao projeto e fazer *push* das alterações.

Git - Exercício 2

No GitHub encontrar um repositório adequado, fazer *fork* do repositório, clonar localmente, criar um novo branch de funcionalidade, modificar um ficheiro e fazer commit e push das alterações.

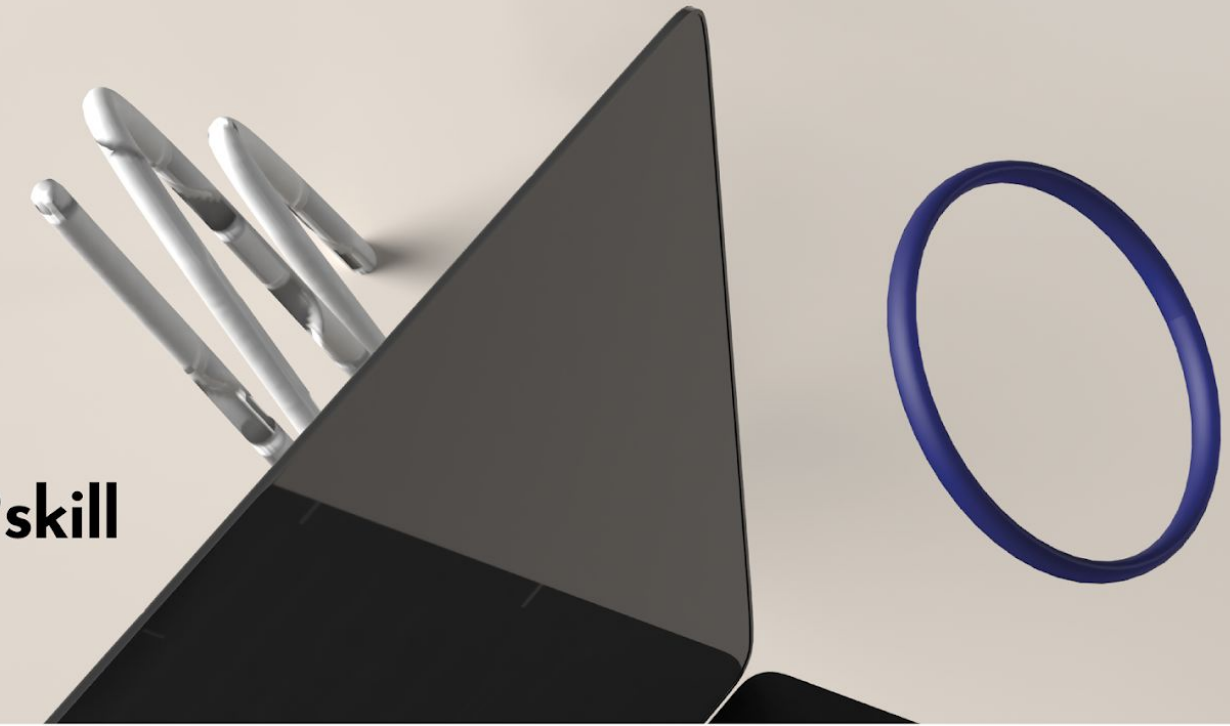
Git - Exercício 3

Utilizar as funcionalidades nativas de Git do IDE para fazer commit do nosso projeto do Rogue para um novo repositório no GitHub.

Git - Recursos de Aprendizagem

- https://learngitbranching.js.org/
- https://git-scm.com/docs

O futuro profissional começa aqui



iscte INSTITUTO UNIVERSITÁRIO DE LISBOA

emprego digital

UPskill