CGRA 2020/2021
Aulas Teórico-Práticas

Informações gerais de funcionamento

v1.3 - 2021.02.11.1700

O funcionamento das aulas teorico-práticas de CGRA envolve o desenvolvimento de trabalhos práticos em grupos de dois estudantes. Neste documento são descritos os procedimentos gerais e as ferramentas associadas às aulas práticas e ao desenvolvimento, submissão e apresentação dos projetos.

Funcionamento das aulas

- **Primeira parte**:
    - Introdução ao tema ou fase do trabalho a focar nessa semana;
    - No caso de apenas ser possível ter parte dos estudantes de uma turma em sala, tentar-se-á, sempre que possível/necessário, transmitir via Discord (ver abaixo) a parte expositiva da aula, para os estudantes que estejam ligados remotamente.
- **Segunda parte**:
    - Dedicada à execução do trabalho por parte dos estudantes, com acompanhamento do docente (e monitor, se disponível).
    - Também nesta parte, os estudantes que estejam remotamente poderão comunicar com o docente ou colegas via Discord.

docente(s) - Discord

- O Discord (https://discord.com) será usado para comunicação entre os estudantes e os docentes

- **Todos os estudantes devem**:

  - **Registar-se no servidor Discord da UC** assim que possível, usando o **link publicado no Moodle**.

  - Ao aceder ao servidor, devem garantir que o seu nickname é o **primeiro e último nome**;

  - Entrar no canal de texto da turma em que estão inscritos, e enviar uma mensagem indicando que pertencem àquela turma, para lhe ser atribuído o *role* correspondente.

- **Durante o horário da aula**, todos os estudantes presentes em sala devem estar ativos no Discord da UC; é conveniente que os restantes estudantes, sempre que possível, também o façam, para:

  - Facilitar a troca de informações entre os estudantes presentes e remotos;

  - Permitir ao docente analisar questões específicas de código, mantendo o distanciamento necessário.

Preparação do repositório Git para os projetos - Gitlab-FEUP

O código desenvolvido ao longo das aulas práticas será mantido num repositório online, alojado no **Gitlab da FEUP**. As instruções para preparar o ambiente de desenvolvimento estão descritas de seguida.
Para saberem mais sobre o uso de **Git e Gitlab** em geral, podem usar os guias básicos

No código base terão um ficheiro “README.md” que utiliza a linguagem ***Markdown***. Podem ler mais sobre Markdown neste link: https://www.markdownguide.org/getting-started/

- Formar **grupos de dois estudantes**, assim que estejam confirmados numa turma (exceções devem ser discutidas com docente das práticas);
- **Fazer login pelo menos uma vez no Gitlab da FEUP** – https://git.fe.up.pt - para serem lá registados;
- **Registar-se na folha de grupos** no Google Drive (link disponível no Moodle);
- **Aguardar por e-mail de atribuição de um repositório** pelos docentes. O repositório estará pré-carregado com o código de base para os trabalhos;
- **Após terem repositório atribuído, devem[1]:**
  - Fazer *clone* do repositório localmente (ver os guias mencionados acima);
  - **NOTA IMPORTANTE 1:** Podem fazer o clone e outros acessos usando SSH ou por HTTPS. **Se usarem acesso por SSH, têm de estar ligados à VPN da FEUP**
  - **NOTA IMPORTANTE 2: garantir que usam as credenciais UP** nesse repositório para os *commits* e *tags*, fazendo, na pasta do repositório:

```
git config user.name “Primeiro Ultimo”
git config user.email “upXXXXXXXXX@fe.up.pt”
```

  - Editar o ficheiro **README.md** da raíz para preencher os dados do grupo (ver guia sobre Markdown acima);
  - Fazer *commit* e criar uma

- Fazer *push* (incluindo *tags* com --*follow-tags*) para o **Gitlab-FEUP;**
- Consultar no Gitlab-FEUP o grafo e confirmar que o *commit* e *tag* dos pontos anteriores foram corretamente registados, e **com o *user* certo**;
- Garantir que ambos os elementos do grupo conseguem fazer localmente no seu computador *fetch/pull/commit/push*, etc. **com o *user* certo**.

Preparação do ambiente de desenvolvimento

Uma parte importante deste primeiro trabalho prático é a preparação do ambiente de desenvolvimento. Devem para isso garantir que têm configuradas as principais componentes necessárias, descritas abaixo, e garantir que conseguem abrir no browser uma aplicação de exemplo.

Componentes necessárias

- **Web Browser com suporte WebGL 2.0**:
  - A aplicação será efetivamente executada através do browser. Uma lista atualizada dos browsers que suportam WebGL 2.0 pode ser encontrada em http://caniuse.com/#feat=webgl2. Atualmente, os browsers Firefox, Google Chrome e Opera têm suporte para esta versão de WebGL, tanto a versão desktop como versão para Android (embora nalguns casos nem todos os dispositivos que podem correr esses browsers tenham capacidades gráficas para executar as aplicações WebGL).
- **A estrutura de base do projeto**:
  - Esta estrutura está disponibilizada no repositório Git fornecido, que deve ser replicado (git clone) localmente por cada estudante

trabalho prático, terá o código-base necessário para iniciarem o desenvolvimento;
    - Por questões relacionadas com restrições de segurança dos browsers, a pasta com esta estrutura deve ser disponibilizada por um servidor web/HTTP (ver ponto seguinte).

- **Servidor HTTP**:
    - Sendo as aplicações acessíveis via browser, e dadas as restrições de segurança dos mesmos que impedem o acesso a scripts através do sistema de ficheiros no disco local, é necessário que as aplicações sejam disponibilizadas através de um servidor web HTTP. No contexto de CGRA, não haverá a geração dinâmica de páginas (as componentes dinâmicas são corridas no interpretador de Javascript do browser), pelo que qualquer servidor HTTP que disponibilize conteúdo estático servirá. Existem várias soluções possíveis para este requisito, incluindo:
        - **Usar um servidor web no próprio computador**: Nalguns casos os estudantes têm já um servidor web (ex: Apache, Node.js) a correr para suportar outros projetos. O mesmo pode ser usado para este efeito, desde que o servidor disponibilize a pasta com a aplicação através de um URL acessível por HTTP. No caso de não terem nenhum servidor, podem correr um mini-servidor usando uma das seguintes opções:
            - **Web server for Chrome:** Um mini-servidor web que corre no próprio Google Chrome: https://chrome.google.com/webstore/detail/web-server-for-chrome/ofhbbkphhbklhfoeikjpcbhemlocgigb
            - **Python**: Caso o Python esteja instalado, pode ser criado um servidor

versão de Python):
    - `python -m SimpleHTTPServer 8080` (para versões 2.x)
    - `python -m http.server 8080` (para versões 3.x).
- **Node.js:** https://www.npmjs.com/package/http-server
- **Outros servidores alternativos:**
    - https://mongoose.ws/
    - http://nginx.org/en/download.html
    - https://www.ritlabs.com/en/products/tinyweb
    - https://caddyserver.com/download
    - https://www.vercot.com/~serva/
    - https://aprelium.com/abyssws/

- **Usar a área web de estudante da FEUP**: colocar o projeto uma pasta dentro da pasta public_html da conta do estudante, e acedendo à mesma através do endereço público **https://paginas.fe.up.pt/~login/mytest** (login será o código de estudante, upXXXX). Neste caso, ficará por omissão acessível a todos (o que pode ser contornado, p.ex. com um ficheiro de controlo de acesso *.htaccess*). Tem também a desvantagem de implicar a edição/atualização dos ficheiros no servidor da FEUP, e obrigar a uma ligação à rede da FEUP para poder editar/carregar a aplicação

- **Um editor de texto ou IDE**: O código que compõe as aplicações será escrito em JavaScript, e armazenado em ficheiros de texto. Para a sua edição existem várias alternativas também:
    - O **Visual Studio Code** é um

recomendada.

- O próprio **Google Chrome** disponibiliza nas suas **"Developer Tools" (Ctrl-Shift-I)** um debugger de JavaScript, que permite fazer execução passo-a-passo, análise de variáveis, consulta da consola, etc. ao código que está a ser corrido no browser, e permite também mapear os ficheiros acessíveis por HTTP aos ficheiros originais armazenados em disco. Pode por isso ser usado como editor e debugger, nomeadamente para quem não queira instalar um IDE específico.
- Qualquer editor de texto pode servir para editar os ficheiros. No entanto, sugere-se fortemente um editor de texto que suporte pelo menos uma estrutura de projeto com navegação numa árvore de ficheiros, para permitir alternar facilmente entre os diferentes ficheiros que constituirão o projeto (Ex: WebStorm, Brackets, Sublime, Atom, Notepad++, …).

Teste do ambiente de desenvolvimento

Nesta fase deve ter já a pasta “tp1”, com os ficheiros do código base disponíveis no repositório do Gitlab, partilhada através de um servidor web. Deve por isso ter também o endereço URL através do qual a pasta é acessível **(NOTA: evitar pastas com espaços e acentos!)**. Abra o browser e direcione-o para o URL referido, e ao fim de alguns segundos deve surgir a aplicação de exemplo, podendo manipular o ponto de vista com o rato, usando o **botão esquerdo para rodar a cena**, o **botão direito para a deslocar lateralmente**, e carregando no **botão central (ou Ctrl+botão esquerdo) para aproximar/afastar**.

Notas sobre o desenvolvimento dos trabalhos

Chegados a este ponto, estarão aptos a

- Os enunciados dos trabalhos serão **publicados no Moodle**;

- **O código de base tem uma estrutura pré-definida que tem de ser respeitada**. Podem ser acrescentados ficheiros e sub-pastas, mas **os fornecidos inicialmente têm de ser mantidos nas localizações originais**;

- Nessa estrutura, existe um ficheiro ***README.md*** na raíz, e outro na pasta de cada TP.

    - O ficheiro da raíz deve ter a identificação do grupo e alguma informação geral que considerem relevante, e deve ser preenchido logo de início.

    - O ***README.md*** de cada TP deve ser usado como registo (log) da execução do trabalho, onde colocam as vossas principais observações e dificuldades, de forma sucinta. DEvem ir atualizando o mesmo conforme vão desenvolvendo o trabalho

- Encontrarão referências nos enunciados a pontos de desenvolvimento que **devem ser assinalados por *tags*** no *Git*. **Devem respeitar escrupulosamente essas indicações**, nomeadamente relativamente ao nome das *tags*;

- **Será analisada a frequência, quantidade, relevância e volume de contribuições dos elementos do grupo**, e em caso de dúvidas sobre o equilíbrio do esforço, serão contactados para esclarecimentos. Nesse sentido, sugere-se que seja feita **uma divisão de tarefas equilibrada, e que sejam feitos *commits* frequentes e com descrições claras** (mas sucintas);

- Sugere-se também o uso de *branches* para o desenvolvimento paralelo de funcionalidades, não esquecendo a importância do *merge* respetivo;

- A gestão do repositório fica ao critério de cada grupo, mas sugerimos fortemente que tenham o *branch* principal (master) atualizado com a última versão funcional do trabalho, e que usem *branches* para desenvolvimento. Sem outra informação, assumiremos o *master* como o oficial

CGRA - Funcionamento TP's

Actualização automática a cada 5minutos

---

[1] Caso usem um IDE com suporte para Git, como o VSCode, poderão fazer grande parte destas operações através da sua interface.