Estructura de directorios en `dune-pdelab-tutorials`

C++ Review Dune

```
work
├── BSD
├── c++
│   ├── CMakeLists.txt
│   ├── doc
│   ├── exercise
│   └── slides
├── CC-BY-SA
├── cmake
│   └── modules
├── CMakeLists.txt
├── config.h.cmake
├── COPYING
├── dune.module
├── dune-pdelab-tutorials.pc.in
├── gridinterface
│   ├── CMakeLists.txt
│   ├── exercise
│   └── slides
├── latexstyle
│   └── exercise.sty
├── LICENSE.md
├── overview
│   ├── abstractions.bib
│   ├── abstractions.tex
│   ├── CMakeLists.txt
│   ├── debug.opts
│   ├── exercise.sty
│   ├── exercise-workflow.tex
│   ├── overview.tex
│   └── release.opts
├── README.md
├── stamp-vc
├── tutorial00
│   ├── CMakeLists.txt
│   ├── doc
│   ├── exercise
│   ├── slides
│   └── src
├── tutorial01
│   ├── CMakeLists.txt
│   ├── doc
│   ├── exercise
│   ├── slides
│   └── src
├── tutorial02
│   ├── CMakeLists.txt
│   ├── doc
│   ├── slides
│   └── src
├── tutorial03
│   ├── CMakeLists.txt
│   ├── doc
│   ├── exercise
│   ├── slides
│   └── src
├── tutorial04
│   ├── CMakeLists.txt
│   ├── doc
│   ├── exercise
│   ├── slides
│   └── src
├── tutorial05
│   ├── CMakeLists.txt
│   ├── doc
│   ├── exercise
│   ├── slides
│   └── src
├── tutorial06
│   ├── CMakeLists.txt
│   ├── doc
│   ├── exercise
│   ├── slides
│   └── src
├── tutorial07
│   ├── CMakeLists.txt
│   ├── doc
│   ├── exercise
│   ├── slides
│   └── src
├── tutorial08
│   ├── CMakeLists.txt
│   ├── doc
│   └── src
├── tutorial09
│   ├── CMakeLists.txt
│   ├── exercise
│   ├── slides
│   └── src
├── TUTORIALLIST
└── workflow
    ├── CMakeLists.txt
    ├── exercise
    └── slides
```

Los ejercicios del repositorio `dune-pdelab-tutorials` se desarrollarán a través de GitPod, una aplicación de Kubernetes de código abierto para entornos de desarrollo automatizados y listos para utilizar fuera de la caja, que nos otorga 100 horas gratuitas de uso durante cada mes. El hipervisor KVM que emplea emula un sistema operativo basado en Arch Linux con el módulo `dune-pdelab` instalado. En este entorno, tendrá un directorio `work` en su carpeta de usuario `gitpod`, que tiene la estructura mostrada a la izquierda. Estos ejemplos están en el directorio `~/work`.

Dune usa `cmake` como sistema de construcción. En `cmake`, hay una clara separación entre el *directorio fuente*, que generalmente está bajo control de versiones (aquí: el subdirectorio `work`) y el *directorio de compilación*, donde se construyen los ejecutables, los archivos de salida. El directorio de compilación será `build`.

El directorio `build` refleja la estructura del directorio fuente. Debería navegar por defecto al directorio `build` del ejercicio actual y trabajar allí. Cada directorio de este módulo corresponde a un tutorial (`tutorial[0-9]`). Todos comparten la siguiente estructura: el directorio `src` contiene el código de ejemplo que se mostró en la conferencia. El directorio `doc` contiene las fuentes de LaTeX de una explicación detallada de el tutorial. El directorio `exercise`, que es relevante para este curso, se subdivide aún más: `task` contiene el código esqueleto sobre el que trabajar durante el ejercicio, `doc` contiene la fuentes de la hoja de ejercicios y `solution` contiene lo que esperas obtener.

En caso de que no esté familiarizado con un sistema GNU/Linux, aquí hay una pequeña lista de comandos frecuentes para el desarrollo de los ejercicios:

- `yay` actualiza la base de datos de los repositorios e instala paquetes actualizados.
- `man <command>` ver las páginas del manual del comando, en caso de estar disponible, escriba `q` para salir. También es válido `<command> --help`.
- `tldr <command>` ver las páginas del manual de la comunidad.
- `mkdir <name>` crea una carpeta `<name>` sin contenido en el espacio de trabajo actual.
- `cmake <path/to/CMakeLists.txt>` crea una carpeta `<name>` sin contenido en el espacio de trabajo actual.
- `cd <dir>` cambia el directorio de trabajo actual a `dir` (al directorio del usuario, si se omite).
- `ls` lista el contenido del directorio de trabajo actual.
- `pwd` imprime el directorio de trabajo.
- `g++ <options> <sources>` compila fuentes de C++ (solo se necesita en el ejercicio de C++)
- `make <executablename>` (re)compila ejecutables en el directorio de compilación actual. Si se omite el nombre del ejecutable, se compilan todos los ejecutables del directorio actual.
- `paraview` es un programa de visualización para archivos VTK.

Recursos

Sección de tutoriales `https://wiki.archlinux.org/title/Bash_(Espa%C3%B1ol)`.

Visores DICOM y renderizado de volumen `https://wiki.archlinux.org/title/List_of_applications/Science#DICOM_viewers_and_volume_rendering`

Pasos en GitPod

1. Iniciar sesión en GitHub y en GitPod.
2. Crear un fork de `https://github.com/cpp-review-dune/prueba-gitpod` (haga click en el botón fork en la esquina superior derecha).
3. Ingresar a la liga `https://github.com/<usuario>/prueba-gitpod`.
4. En la barra de direcciones ingrese a `gitpod.io/#https://github.com/<usuario>/prueba-gitpod`.

[1] Adaptación de *Structure of the course material for the IWR Dune Course, March* 2021.