

A top-down view of a person's hands typing on a silver laptop. The laptop screen displays a presentation slide with a pink and green abstract background. The slide contains the text: 'We stand on', 'CREATIVE CAMPAIGNS', 'Presentations are tools.', 'UNIQUE CONCEPTS', 'Presentations are tools.', 'DIRECT BUSINESS', 'IMPACT', 'Presentations are tools.', and 'RESEARCH'. A large yellow banner with the text 'Conda Crash Course' is overlaid on the image. The banner has a white rectangular area in the center containing the 'PYTHON DATA SCIENCE MASTERY' logo and text. The background is a white marble desk with various items like glasses, a mouse, and a small plant.

Conda Crash Course



PYTHON DATA SCIENCE MASTERY

¿QUÉ ES CONDA?

- Conda es una **herramienta de línea de comandos** usada para gestionar entornos y paquetes
- Gestiona automáticamente las dependencias de los paquetes que instalamos
- **También es un repositorio de paquetes.** Es decir, muchos paquetes de Python están disponibles online en el repositorio de Conda para ser instalados a demanda
- Los paquetes son diferentes por plataforma y sistema operativo. Pero **Conda identifica automáticamente los que tiene que instalar en nuestro equipo**
- Incluso identifica la versión de Python e instala la versión del paquete que sea compatible
- La estructura básica de un comando Conda es: *conda COMMAND OPTIONS --SWITCH*
- Los comandos Conda se ejecutan en el terminal
- Para ver los más frecuentes: *conda --help*

GESTIÓN DE ENTORNOS

- Los entornos permiten tener instalados y aislados diferentes versiones de paquetes e incluso del propio Python
- Conda permite crear, mantener y cambiar fácilmente entre entornos
- **El entorno por defecto se llama base o root**
- En el terminal ya vemos el entorno, que aparece antes del prompt entre paréntesis

Ver los entornos disponibles en el sistema

- *conda env list*
- El que está activo aparece con un asterisco
- También aparece el directorio de cada entorno
- El directorio es simplemente donde están los recursos de ese entorno
- No tiene nada que ver con el directorio activo de un proyecto
- De hecho, varios proyectos pueden compartir el mismo entorno

GESTIÓN DE ENTORNOS

Ver los paquetes instalados en el entorno activo

- *conda list*

Ver los paquetes instalados en otro entorno

- *conda list --name nombre_entorno*

Crear un entorno:

Lo básico es el nombre del entorno. Opcionalmente se pueden especificar la versión de python, los paquetes iniciales e incluso sus versiones.

- *conda create --name nombre_entorno python=3.10 pandas=1.0 numpy scipy*

Cambiar entornos

- Para activar un entorno: *conda activate nombre_entorno*
- Para desactivar un entorno: *conda deactivate* (te devuelve al base)

Eliminar un entorno

- *conda env remove --name nombre_entorno*

GESTIÓN DE ENTORNOS

Exportar un entorno a un archivo

Como convenio se le llama `environment.yml`, aunque puedes ponerle el nombre que quieras.

- `conda env export -n nombre_entorno --file environment.yml` (también vale `-f`)

También se podría crear el archivo a mano, usando la estructura de un `yml` de ejemplo y poniendo sólo la versión de python y los paquetes principales, y luego al instalarlo Conda ya se ocuparía de instalar todas las dependencias necesarias.

Importar un entorno de un archivo

Podemos crear un entorno a partir de un `yml`, que tiene que estar en el directorio actual de la sesión. Usaremos:

- `conda env create --file nombre_entorno.yml --name nombre_entorno`

GESTIÓN DE PAQUETES

Instalar paquetes (se instalarán en el entorno activo)

- Uno solo: *conda install numpy*
- Varios a la vez: *conda install numpy pandas*
- Una versión determinada: *conda install pandas=1.0*

Ver paquetes instalados

- *conda list*

Actualizar paquetes (busca la versión más reciente que sea compatible con otros paquetes)

- *conda update pandas*

Eliminar paquetes

- *conda remove numpy pandas*

GESTIÓN DE PAQUETES - CANALES

- Los canales son como **grupos dentro del repositorio conda**
- **El canal oficial es el main**, y es el que se usa por defecto si no especificas nada
- Todos los paquetes que están en el main ha sido revisados por el equipo de Anaconda
- **Después está conda-forge**, que es como el canal de Conda **mantenido por la comunidad**.
Hay muchos más paquetes
- Y luego los individuales. Una persona u organización puede tener su propio canal en el que accederás a sus paquetes
- Se busca un canal con: `conda search --channel nombre_canal`
- En lugar de --channel se puede usar -c
- Ej: `conda search -c conda-forge` nos dirá todos los paquetes disponibles en conda-forge
- **Hay veces que buscamos un paquete pero no sabemos en qué canal está**. Podemos usar:
 - `conda search nombre_paquete` y nos dirá los canales en los que está disponible

Instalar un paquete de un canal

- `conda install -c nombre_canal nombre_paquete`

FLUJO DE TRABAJO QUE USAREMOS NORMALMENTE

1. Abrir Anaconda Prompt o Terminal
2. Revisar los entornos disponibles: `conda env list`
3. Activar el que queremos usar para la sesión: `conda activate "nombre_entorno"`

4. Ir al directorio del proyecto: `cd "ruta_directorio"`
5. Lanzar Jupyter: `jupyter notebook`
6. Entrar al notebook concreto y comenzar a trabajar



*Durante la primera
fase del curso
usaremos estos pasos*

PROBLEMA HABITUAL

Compartir un `environment.yml` no garantiza versiones exactas de todas las dependencias, no es 100% reproducible

Distintos equipos = versiones diferentes = errores y comportamientos inesperados

Necesitamos por tanto una herramienta capaz de congelar las versiones y multiplataforma:

- **Conda-lock**

Esta herramienta reproduce entornos idénticos en cualquier sistema operativo, además de soportar librerías instaladas con conda y pip, lo que la hace excelente para trabajar en equipo o en producción.

Crearemos el entorno **`pds_m_foundations`**