Curso Node.js

Instalación e introducción a **npm**

Instalar node.js

- Node.js es un proyecto *open source* que se actualiza **muy a menudo**

- https://nodejs.org

- Versión más reciente (a fecha 28/9/2015): **4.1.1** (23/9/2015)

Instalar node.js

- No es recomendable instalar la más reciente para producción
- Versiones **pares** se consideran estables
- Versiones **impares** inestables
- https://nodejs.org/en/download/releases/

Instalar node.js

- Vamos a utilizar en este curso la última versión de la serie **0.12**: 0.12.7, del 9 de Julio de 2015.

- https://nodejs.org/dist/v0.12.7/

- Docs de la versión en https://nodejs.org/docs/latest-v0.12.x/api/

Instalar node.js

- Para instalar en Windows, Mac: **instaladores y binarios**
- Para instalar en Linux:
  - Vía gestor de paquetes
  - Binarios
  - Source

Instalar node.js

- Instalar con gestor de paquetes (Debian, Ubuntu, Enterprise Linux, Fedora, Gentoo, openSUSE / Suse Linux Enterprise)

- https://nodejs.org/en/download/package-manager/

Instalar node.js

- Instalar con binarios: descargar tarball (32/64 bits), crear enlaces simbólicos

➡ `wget http://nodejs.org/dist/v0.12.7/node-v0.12.7-linux-x64.tar.gz`

➡ `sudo tar -C /usr/local --strip-components 1 -xzf node-v0.12.7-linux-x64.tar.gz`

- Esto deja los ejecutables en /usr/local/bin

Instalar node.js

- Comprobar instalación
  - node -v
  - npm -v

npm

- Gestor de paquetes y dependencias para node (CLI)

- Registro público de paquetes (https://npmjs.com)

- Alrededor de 200.000 paquetes disponibles, open source: herramientas de línea de comandos, librerías, módulos, plugins, etc.

- Documentación https://docs.npmjs.com/

npm

- Dos formas de instalar paquetes: global y local
- Instalación global: paquetes (o comandos) disponibles en todo el sistema
- Instalación local: paquetes disponibles en el directorio actual

npm

- npm utiliza el directorio ./*node_modules* para guardar los paquetes instalados localmente

- npm utiliza el directorio */usr/local/bin* para guardar los paquetes instalados globalmente (puede requerir permisos de root)

- Se puede modificar la ruta por defecto donde guarda los paquetes globales

npm

- Instalar dependencias locales (en ./node_modules)

➡ `npm install <package_name>`

- Instalar paquetes globalmente:

➡ `npm install **-g** <package_name>`

npm - install

- **npm install** (alias npm i) acepta como parámetro:
  - Un nombre de paquete en npmjs.org
  - Opcionalmente, una versión específica
  - npm install lodash@latest (la más reciente)
  - npm install react@0.13.x (la última revisión de 0.13)

npm - uninstall

- **npm uninstall** desinstala un paquete

- **npm update** actualiza un paquete instalado

npm - package.json

- Un archivo en formato JSON que incluye:
- Datos del proyecto (nombre, autor, url, licencia…)
- Comandos o tareas (clave **scripts**)
- Lista de dependencias (claves **dependencies** y **devDepedencies**)
- https://docs.npmjs.com/files/package.json

npm - package.json

```
{
  "name": "cursoreact",
  "version": "1.0.0",
  "description": "",
  "main": "index.js",
  "scripts": {
    "test": "./node_modules/.bin/mocha"
  },
  "author": "cdelaorden",
  "license": "MIT",
  "dependencies": {
    "babel": "^5.8.23",
    "bluebird": "^2.10.0",
    "eventemitter3": "^1.1.1",
    "mori": "^0.3.2",
    "page": "^1.6.3",
    "react": "^0.13.3"
  },
  "devDependencies": {
    "babel-core": "^5.8.23",
    "babel-loader": "^5.3.2",
    "istanbul": "^0.3.20",
    "jsdom": "^3.1.2",
    "mocha": "^2.3.2",
    "rewire": "^2.3.4",
    "should": "^7.1.0",
    "sinon": "^1.16.1",
    "statik": "^1.4.6",
    "webpack": "^1.12.0"
  }
}
```

npm - package.json

- Para crear un nuevo package.json (con opciones por defecto):

➡ `npm init`

npm - package.json

- Una vez tenemos un archivo **package.json** en nuestro directorio, podemos **guardar** las dependencias que instalemos:

➡ `npm install package —save`

➡ `npm install package —save-dev`

npm - package.json

- Instalar una versión específica de un paquete

➡ `npm install —save react@0.13`

- Desinstalar una dependencia

➡ `npm uninstall react`

➡ `npm uninstall react —save`

npm - package.json

- El archivo package.json nos permite distribuir o guardar nuestro código en un repositorio con la lista de dependencias, por lo que **no es necesario incluir la carpeta node_modules**

- Para instalar **todas las dependencias especificadas en package.json**

➡ `npm install`

npm - package.json

- Tenemos dependencias de producción (guardadas con —save)

- Tenemos dependencias de desarrollo (guardadas con —save-dev)

- Para instalar **sólo las dependencias de producción**

➡ `npm install —production`

npm - package.json

- ¿Qué son las dependencias de desarrollo?
- Normalmente, utilidades que sólo se utilizan en tiempo de desarrollo:
- Test runners, empaquetadores, etc.

npm - scripts en package.json

- Podemos guardar comandos de CLI en scripts en package.json

- Y ejecutar con **npm run <nombre>**

- El script especial **start** se ejecuta directamente con **npm start**, y el script especial **test** se ejecuta directamente con **npm test**

npm - scripts

```
"scripts": {
    "test": "./node_modules/.bin/mocha",
    "start": "node -v",
    "hola": "echo 'Hola mundo!'"
  },
```

- **npm test** - intentará ejecutar mocha (instalado globalmente)
- **npm start** - devolverá por consola la versión actual de node
- **npm run hola** - imprimirá en consola el texto “Hola mundo!”

nvm

- Una serie de scripts de bash que nos permiten tener múltiples versiones de node.js y npm instaladas

- https://github.com/creationix/nvm

nvm

- Una vez instalado, nos permite instalar y desinstalar directamente versiones de node con
  - nvm install 0.10
  - nvm install 0.12
  - nvm uninstall 0.11

nvm

- Puesto que podemos tener varias versiones, podemos elegir una para la sesión/shell actual con:

- nvm use 0.10

- nvm use 0.12

nvm

- Más aún, podemos definir para un proyecto (directorio) específico la versión que queremos, en un archivo **.nvmrc**

- echo 0.12.7 > .nvmrc

- Y después **nvm use** dentro de esa carpeta (leerá la versión del archivo .nvmrc)

- Esto nos permite tener diferentes proyectos para distintas versiones