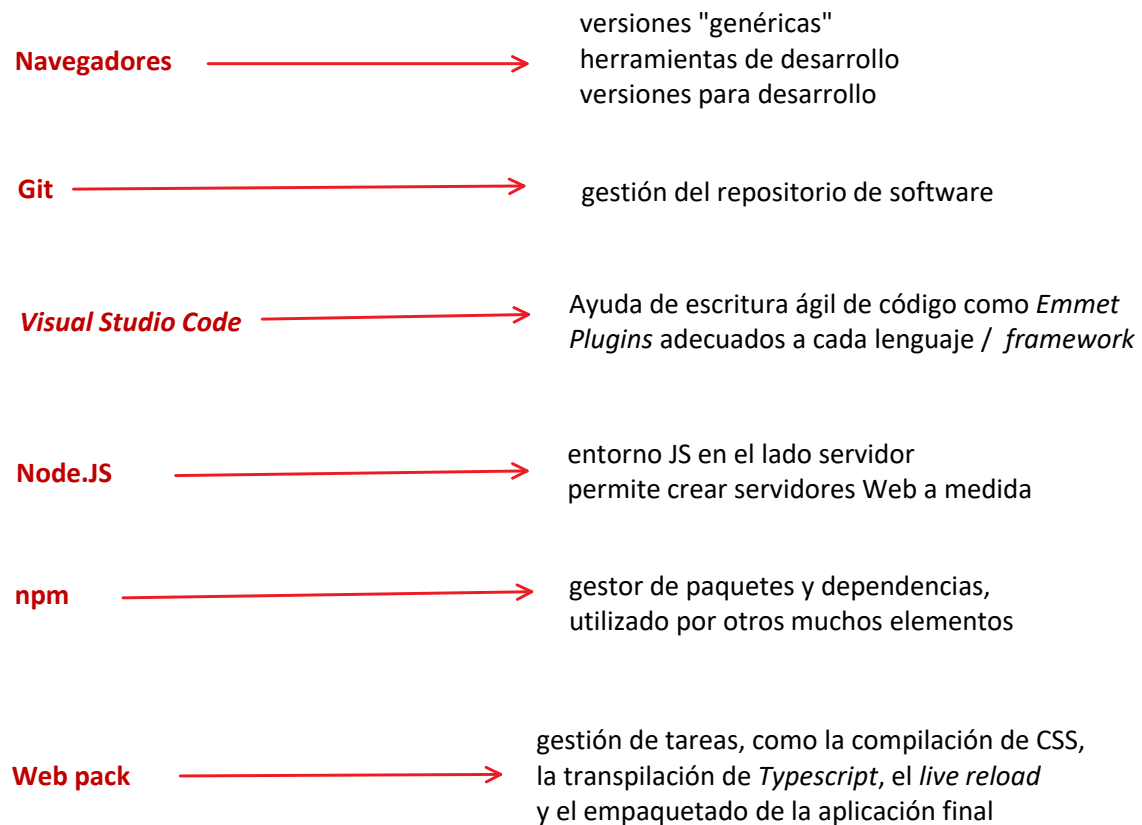


Entorno de trabajo

sábado, 9 de septiembre de 2017 13:33

Navegadores.
Editores de código
Gestión de versiones. GIT
NodeJS y npm. Empaquetado



rundll32 sysdm.cpl,EditEnvironmentVariables

Entornos de desarrollo front

viernes, 12 de octubre de 2018

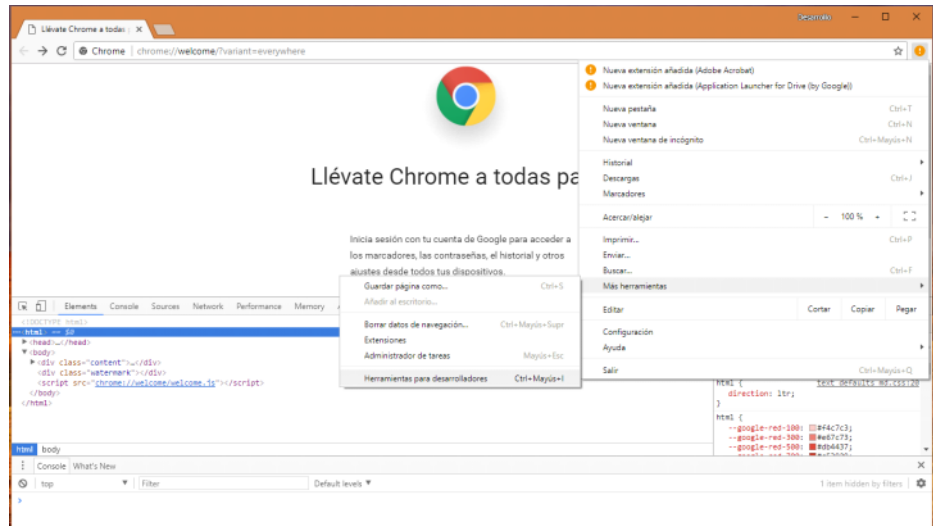
18:35

- Editores de código / IDEs
 - VSC: <https://code.visualstudio.com/>
 - Sublime: <https://www.sublimetext.com/>
 - Atom: <https://atom.io/>
 - Eclipse: <https://www.eclipse.org/ide/>
 - Visual Studio / Oracle IDEs ...
- Gestores de paquetes
 - Node / npm: <https://www.npmjs.com/>
 - yarn: <https://yarnpkg.com>
- Servidor Web de desarrollo
 - IIS
 - Apache / xAMP
 - NodeJS / http-server
- Repositorios de Software
 - Git: <https://git-scm.com/>
 - En la nube
 - GitHub: <https://github.com/>
 - GitLab: <https://about.gitlab.com/>
 - Bitboucket : <https://bitbucket.org/>
- Gestión de Tareas y Empaquetado
 - Grunt: <https://gruntjs.com/>
 - Gulp: <https://gulpjs.com/>
 - Scripts npm: <https://www.npmjs.com/>
 - WebPack: <https://webpack.js.org/>
- Integración continua: DevOps
 - Solano CI,
 - Bamboo,
 - Pipeline,
 - Apache Continuum,
 - Hudson,
 - Jenkins,
 - GoCD,
 - CruiseControl

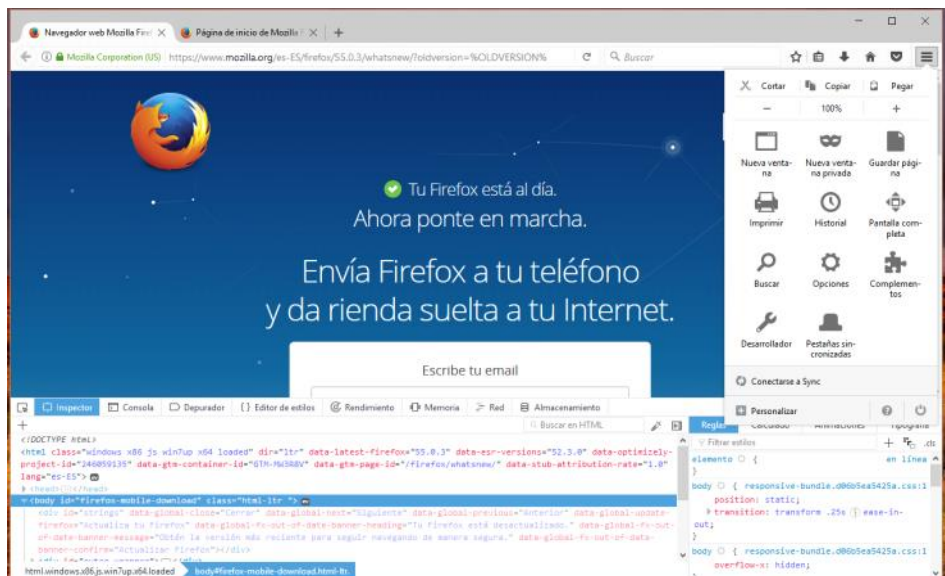
Navegadores

sábado, 9 de septiembre de 2017 18:20

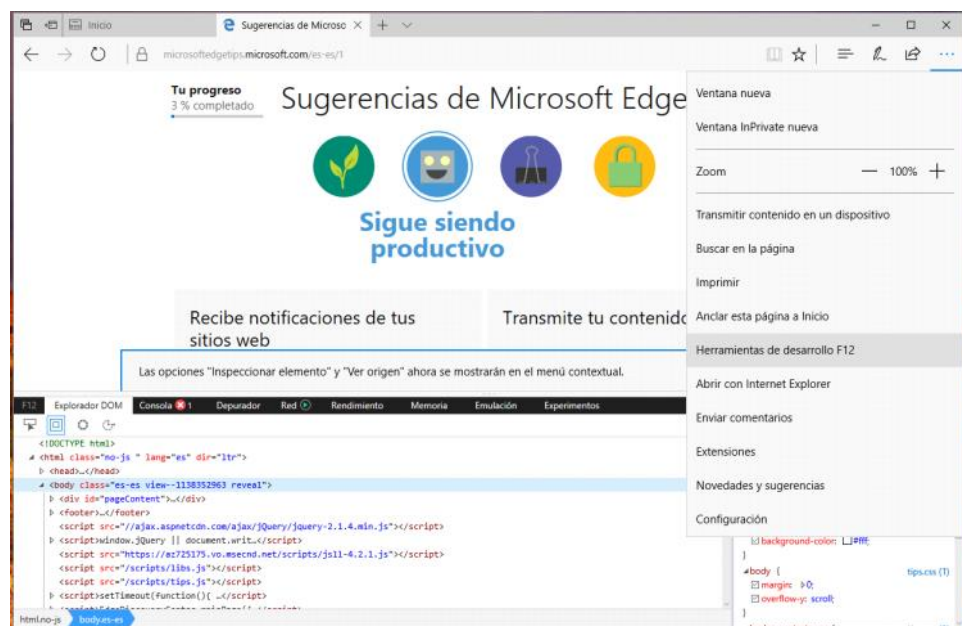
Herramienta de desarrollador en las últimas versiones de Chrome, en este caso la 60.0.3



Herramienta de desarrollador en las últimas versiones de Firefox, en este caso la 55.0.3



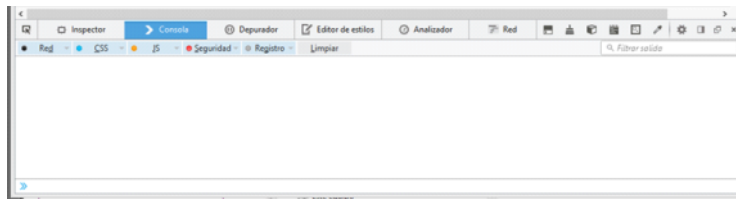
Herramienta de desarrollador en Edge, el navegador incorporado desde Windows 10



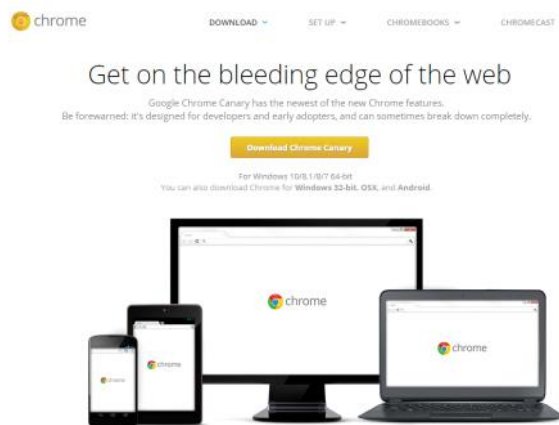
Consola JavaScript

El ambiente en el que se ejecutan los scripts (navegador) proporciona un objeto console, que corresponde a la consola JS que podemos hacer visible en la parte inferior del navegador.

Los métodos console.log y console.dir son otra alternativa para presentar texto en pantalla desde un script. Algunos autores la prefieren a alert(), por considerarla menos intrusiva.



Versiones "especiales" para desarrolladores

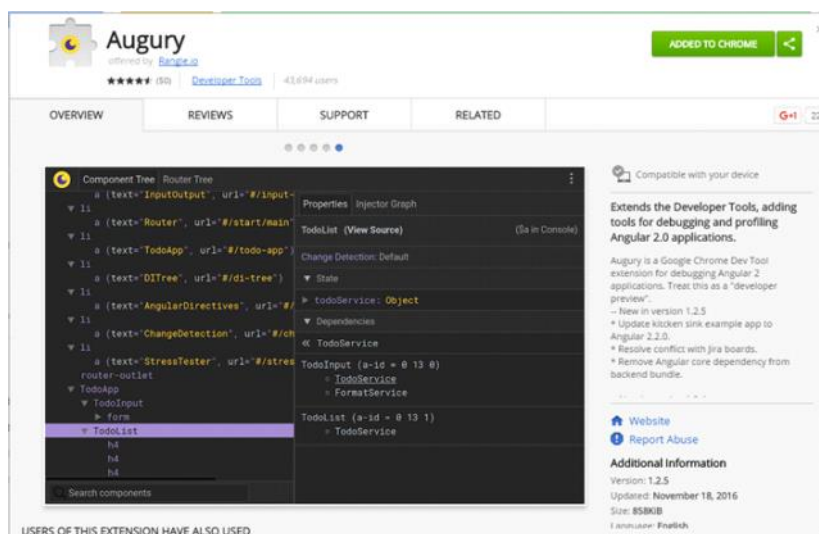


<https://www.google.es/chrome/browser/canary.html>



<https://www.mozilla.org/es-ES/firefox/developer/>

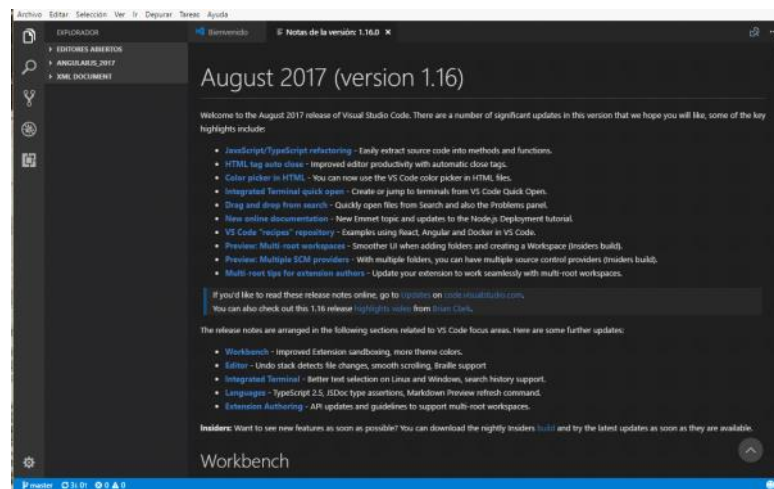
Plugin en Chrome



<https://chrome.google.com/webstore/detail/augury/elgalmkoelokbchhkhacckoklkejhnd>

Editores de código

sábado, 9 de septiembre de 2017 18:21



Visual Studio Code

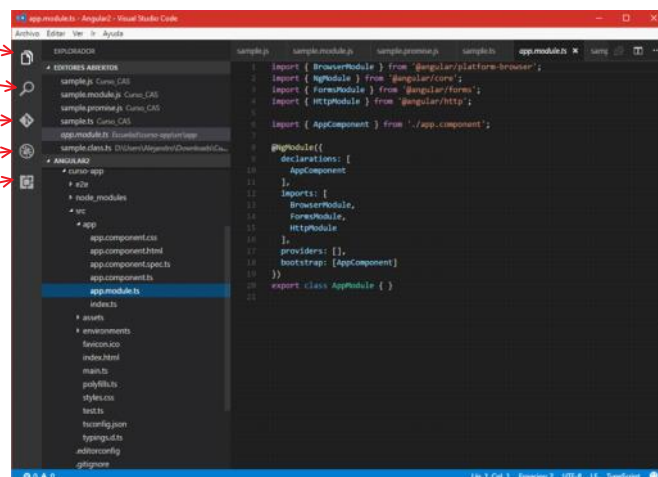
Carpets y ficheros

Búsquedas en el proyecto

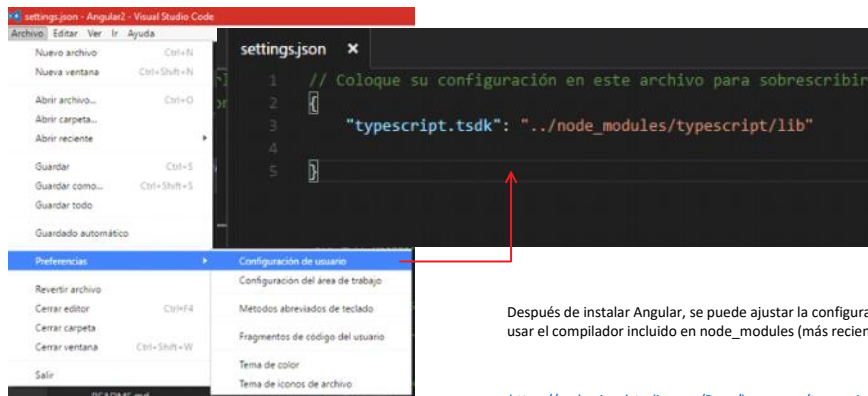
Git - GitHub integrado

Depurador

Extensiones



Configuración VSC

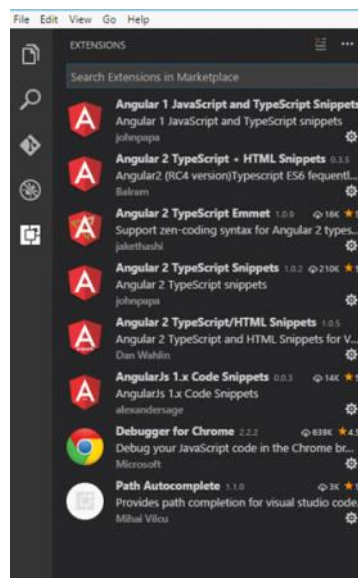


Después de instalar Angular, se puede ajustar la configuración para usar el compilador incluido en node_modules (más reciente)

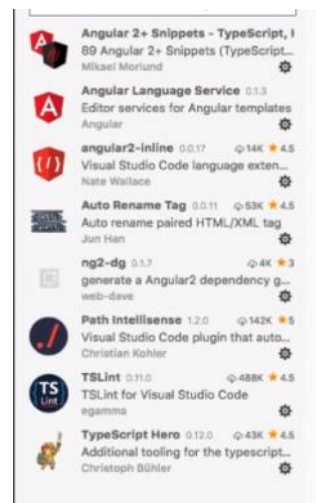
<https://code.visualstudio.com/Docs/languages/typescript>

Extensiones de VSC

- Angular 2 Typescript
- Angular 2 Typescript Emmet
- Angular 4 and TypeScript/HTML VS Code Snippets (Dan Wahlin)
- **Angular 4 TypeScript Snippets ***** (John Papa)**
- **Angular Language Service**
- **angular2-inline (Nate Wallace)**
- **TSLint ***** (egamma)**
- Auto Import (steoates) / TypeScript Hero (Christoph Bühler)
- AutoRenameTag
- **Debugger para Chrome ***** (Microsoft)**
- **npm *** (egamma)**
- **Path Intellisense *** (Christian Kohler)**



A.Basalo, Angular 4



<https://marketplace.visualstudio.com/items?itemName=johnpapa.angular-essentials>

Pack con las extensiones de Angular, según recomendación de John Papa

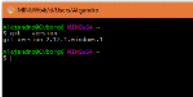
- Angular v5 Snippets
- Angular Language Service
- Angular Inline
- Path Intellisense
- tslint
- Chrome Debugger
- Editor Config
- PrettierVS Code
- Winter is Coming theme



Resultado



Comprobación



Git es un SCV distribuido diseñado para la gestión eficiente de flujos de trabajo distribuido no lineales. Git fue diseñado y desarrollado inicialmente por Linus Torvalds en 2005 para el desarrollo del kernel de Linux. La licencia de Git es libre y hay distribuciones oficiales para los sistemas operativos

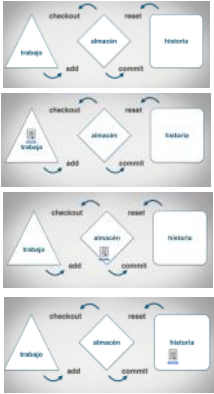
- Mac OS X
- Windows
- Linux y
- Solaris.

La distribución de Git incluye herramientas de línea de comando y de escritorio.

Además, hay disponibles herramientas proporcionadas por terceros que permiten una mayor integración con el escritorio o con entornos de desarrollo.



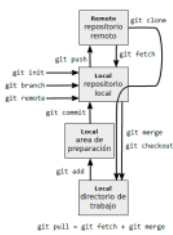
Ciclo de operaciones



Comandos

git init - crear un repositorio local en un directorio.
git clone - crear un repositorio local haciendo una copia de otro (local o remoto)
git add - registra los ficheros del directorio de trabajo cuyos cambios se quieren
git commit - confirma los cambios de los ficheros registrados
git remote - configura los repositorios remotos
git branch - crear y destruir ramas
git checkout - permite cambiar de rama en el directorio de trabajo (Por defecto se trabaja en la rama denominada master)
git push - enviar los cambios a un repositorio remoto en la rama indicada
git pull - actualizar el repositorio con los cambios de un repositorio local

Este comando es esencialmente un encadenamiento de dos comandos:
git fetch - obtiene los cambios de una rama remota
git merge - fusiona si es posible estos cambios con una rama local.



El repositorio creado tiene tres partes diferenciadas. El directorio es la que se llama directorio o espacio de trabajo, y dentro de la carpeta .git que ha sido creada por el comando se encuentra el área de preparación, que actúa como una zona intermedia, y el historial de cambios.

Git permite el uso de ramas (branches).

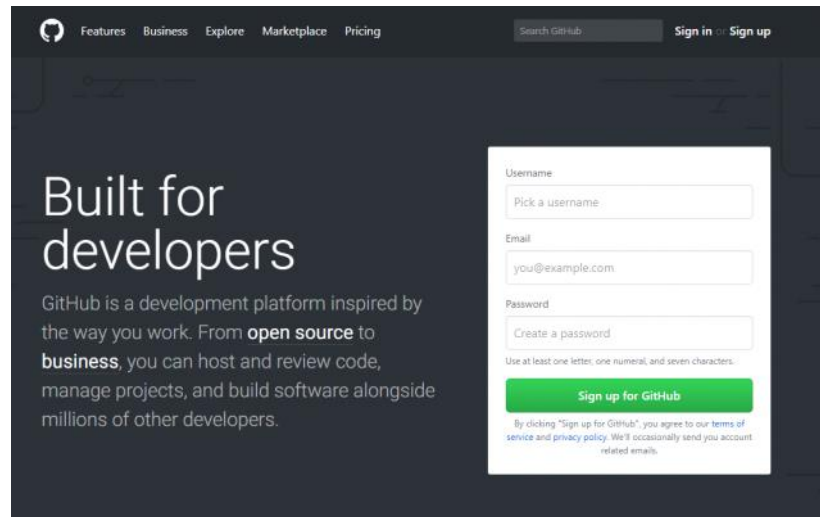
El comando git pull es un ejemplo de la orientación a caja de herramientas que caracteriza el diseño de Git.

Git está implementado como un conjunto de programas y scripts de shell que son fácilmente encadenables para formar nuevos comandos. Además, Git cuenta con mecanismos para lanzar scripts de usuario cuando suceden ciertos eventos en el flujo de trabajo denominados puntos de enganche (hooks).

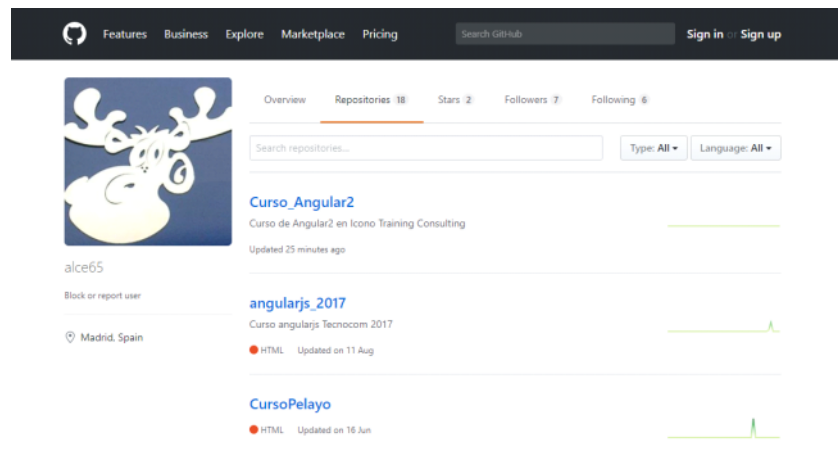
Git en la Nube. GitHub

domingo, 10 de septiembre de 2017 12:28

- **GitHub** →
- GitLab
- Bitbucket



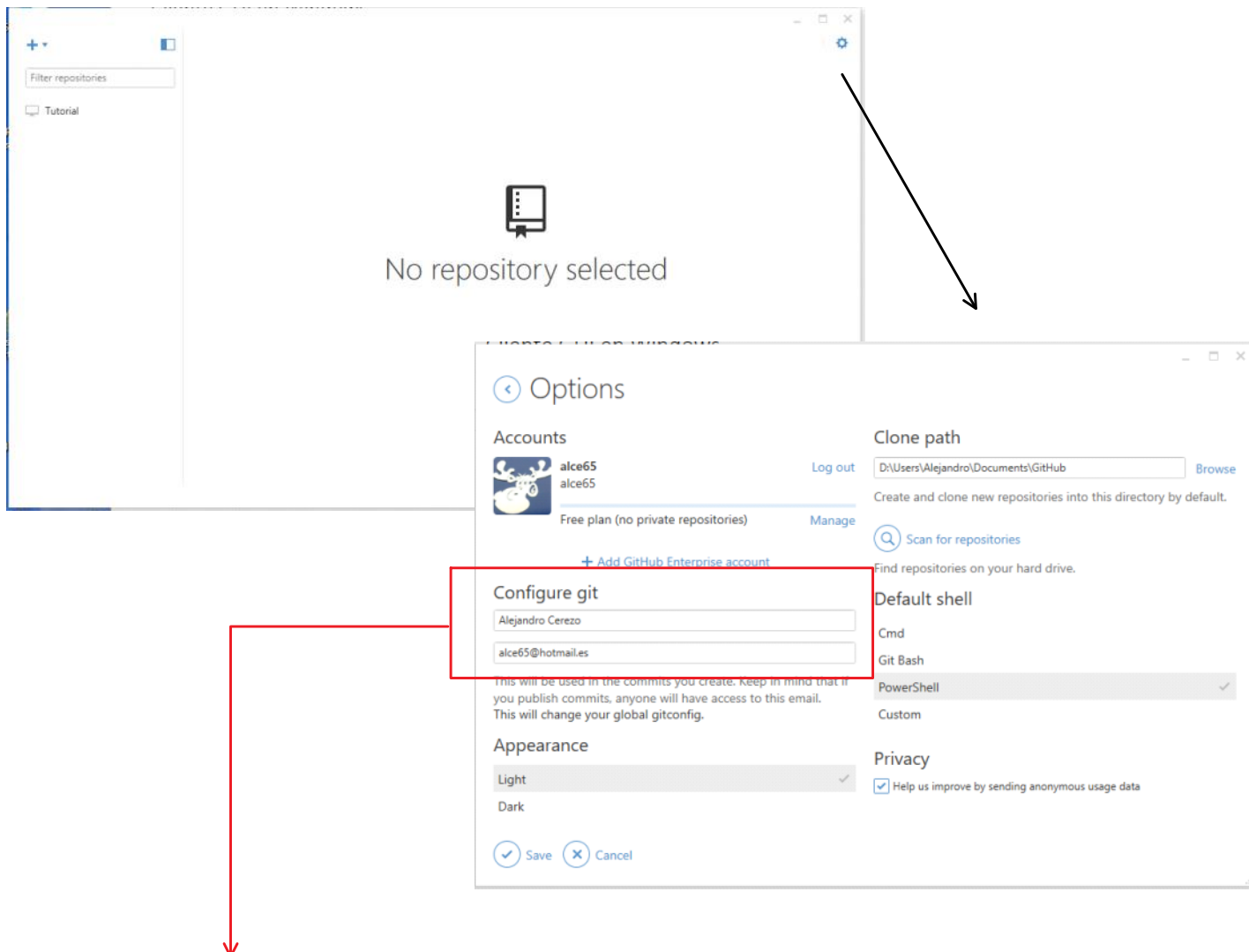
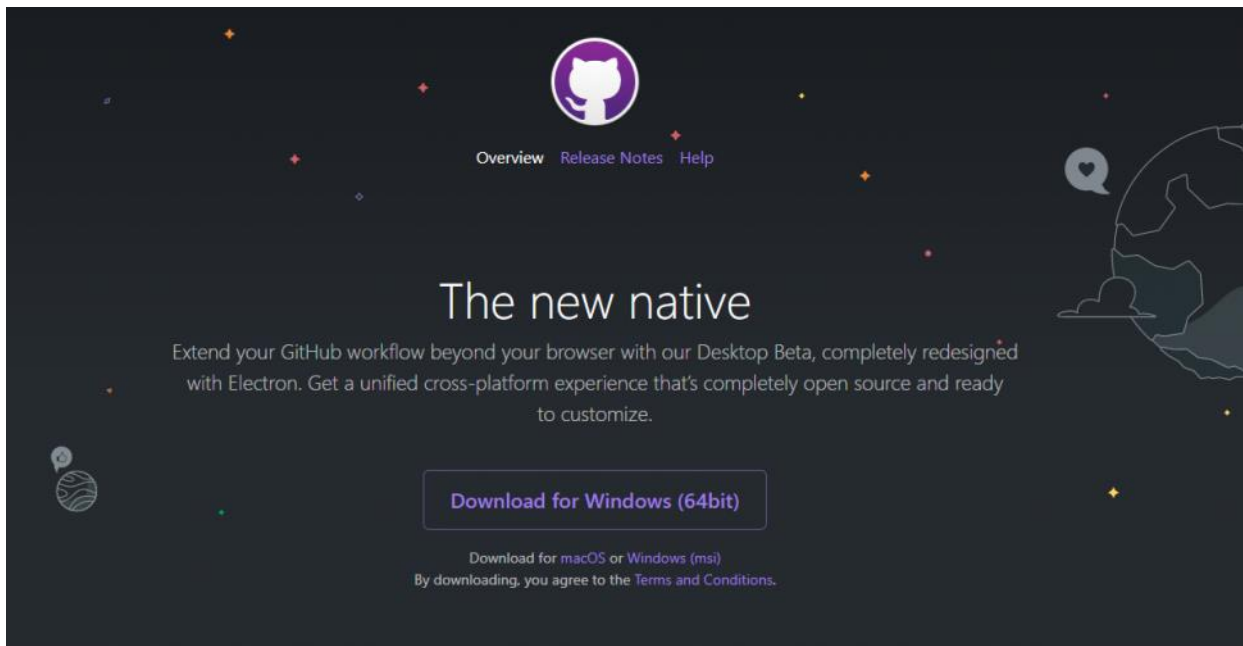
<https://github.com/alce65>

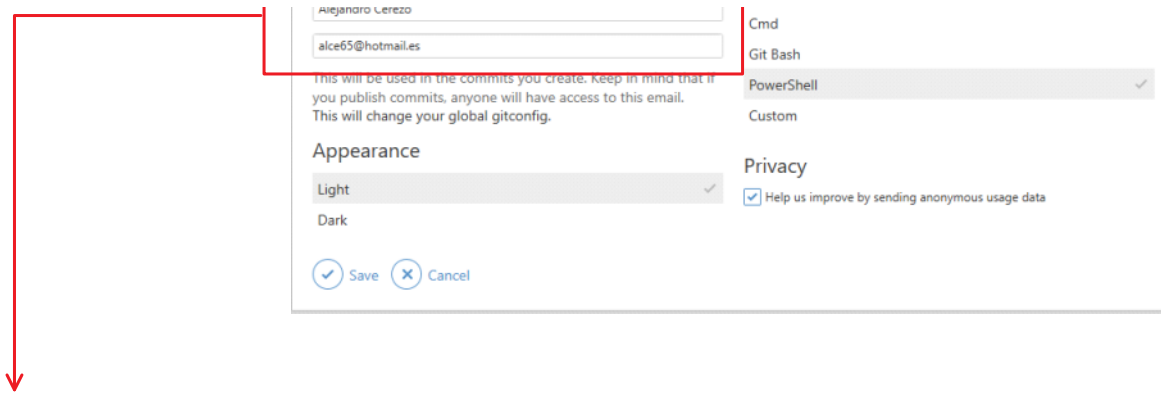


GUI para GitHub

domingo, 10 de septiembre de 2017 12:52

<https://desktop.github.com/>





Corresponde a los comandos de configuración de *git*

```
$ git config --global user.name "Pepe Perez"  
$ git config --global user.email pperez@example.com
```

Repositorio en GitHub

domingo, 10 de septiembre de 2017 12:25

Nuevo repositorio en GitHub

Create a new repository

A repository contains all the files for your project, including the revision history.

Owner: **alice65** / Repository name: **Curso_Angular2** ✓

Great repository names are short and memorable. Need inspiration? How about probable-spork.

Description (optional): **Curso de Angular2 en Icono Training Consulting**

Public: Anyone can see this repository. You choose who can commit.

Private: You choose who can see and commit to this repository.

☒ Initialize this repository with a README
This will let you immediately clone the repository to your computer. Skip this step if you're importing an existing repository.

Add .gitignore: **Node** Add a license: **MIT License**

Create repository

↓

This repository: **alice65 / Curso_Angular2**

Unwatch 1 Star 0 Fork 0

Code Issues 0 Pull requests 0 Projects 0 Wiki Settings Insights

Curso de Angular2 en Icono Training Consulting **Edit**

Add topics

1 commit 1 branch 0 releases 1 contributor

Branch: master New pull request Create new file Upload files Find file **Clone or download**

File	Commit	Time
alice65 initial commit	Initial commit	just now
.gitignore	Initial commit	just now
LICENSE	Initial commit	just now
README.md	Initial commit	just now

README.md

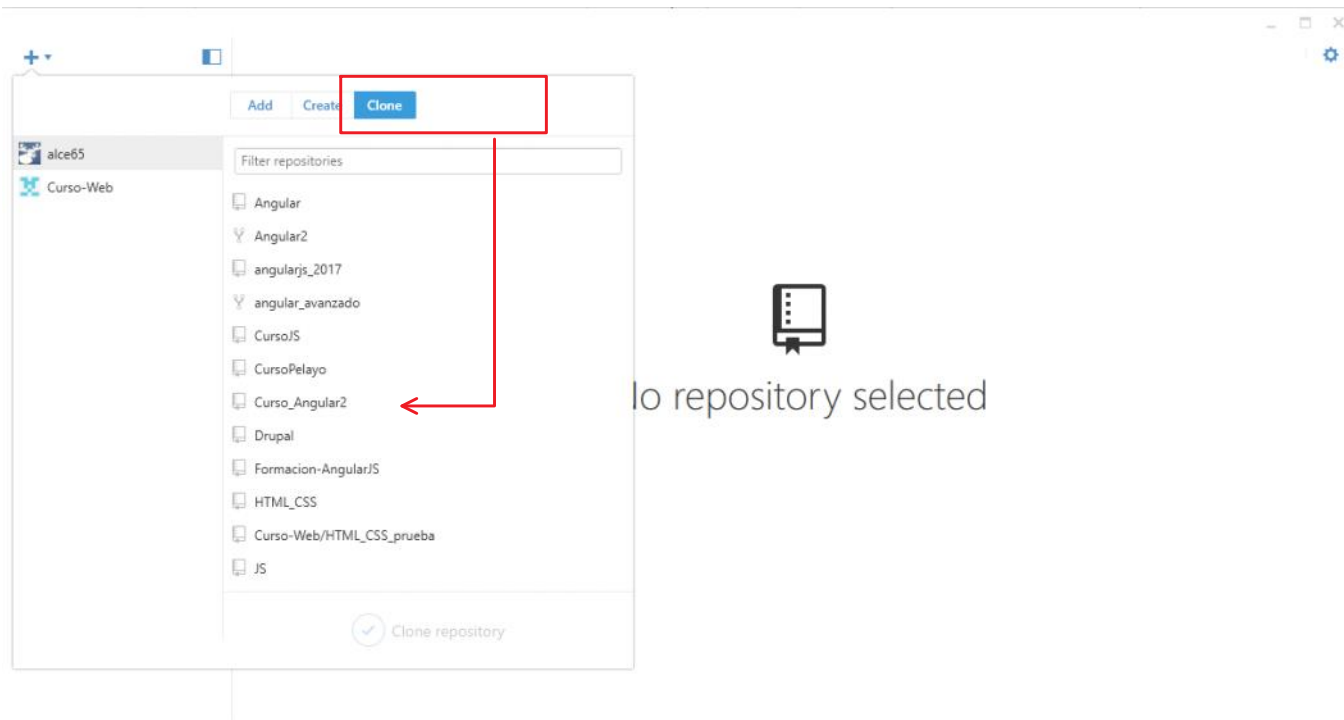
Curso_Angular2

Curso de Angular2 en Icono Training Consulting

Clonación local del repositorio

Clonar un repositorio

domingo, 10 de septiembre de 2017 17:22



Construcción de proyectos / empaquetado



<http://gruntjs.com/>



<http://gulpjs.com/>



<http://broccolijs.com/>



<https://webpack.github.io/>

herramientas para procesar
los fuentes de la aplicación

- Reducción del tiempo de descarga
- Preprocesadores CSS
- Optimización del código, CSS, HTML
- Cumplimiento de estilos y generación de JavaScript (transpilación)

Grunt

miércoles, 3 de mayo de 2017 16:51

```
$:>npm install -g grunt-cli
```

```
D:\Users\Alejandro>npm install -g grunt-cli
D:\Users\Alejandro\AppData\Roaming\npm\grunt -> D:\Users\Alejandro\AppData\Roaming\npm\node_modules\grunt-cli\bin\grunt
grunt-cli@0.1.13 D:\Users\Alejandro\AppData\Roaming\npm\node_modules\grunt-cli
├─ resolve@0.3.1
├─ nopt@1.0.10 (abbrev@1.0.7)
└─ findup-sync@0.1.3 (lodash@2.4.2, glob@3.2.11)
```



GRUNT

<https://gruntjs.com/>

Gulp

sábado, 13 de octubre de 2018 11:00



<https://gulpjs.com/>

```
$:>npm install -g gulp-cli
```

```
const gulp = require('gulp')
const postcss = require('gulp-postcss')
const autoprefixer = require('autoprefixer')
const postcss_preset_env = require('postcss-preset-env')
const precss = require('precss')
gulp.task('post-css', () => {
  const aProcessors = [
    // autoprefixer,
    postcss_preset_env,
    precss
  ]
  return gulp
    .src('./src/*.css')
    .pipe(postcss(aProcessors))
    .pipe(gulp.dest('./dist'))
})
gulp.task('watch', () => {
  gulp.watch('./src/**/*.css', ['post-css'])
})
gulp.task('default', ['post-css', 'watch'])
```


JS en el servidor (SSJS): Node.js

<https://nodejs.org/>



Node.js® es un entorno de ejecución para JavaScript construido con el motor de JavaScript V8 de Chrome. Node.js usa un modelo de operaciones E/S sin bloqueo y orientado a eventos, que lo hace liviano y eficiente. El ecosistema de paquetes de Node.js, npm, es el ecosistema mas grande de librerías de código abierto en el mundo.

Important security releases, please update now!

Descargar para Windows (x64)

Versión LTS →

v6.11.3 LTS Recomendado para la mayoría	v8.4.0 Actual Últimas características
Otras Descargas Cambios Documentación del API	Otras Descargas Cambios Documentación del API

Node.js® es un **entorno de ejecución para JavaScript** (una plataforma de software)

- Se emplea para construir aplicaciones de red escalables (especialmente servidores).
- Está construido con el motor de JavaScript V8 de Chrome
- Utiliza un modelo de operaciones que lo hace liviano y eficiente, gracias a
 - o **operaciones E/S sin bloqueo** y
 - o orientado a eventos, con un **bucle de eventos de una sola hebra**

Además, el **ecosistema de paquetes de Node.js, npm**, es el ecosistema más grande de librerías de código abierto en el mundo.

Su funcionalidad es especialmente adecuada en

- operaciones en tiempo real (e.g. chats)
- Bases de datos *NoSQL* / No relacionales

Node.js: ampliación de JS



Al core de JS no le acompañan las APIS habituales en cualquier lenguaje de programación

Node.js puede entenderse como la ampliación de JS para llegar a ser un lenguaje "completo", independiente de un entorno huésped

v8 (JavaScript)

- Una gramática que define la sintaxis del lenguaje
- Un intérprete/compilador que lo sabe interpretar y ejecutar
- Mecanismos para interactuar con el mundo exterior (llamadas al sistema)
- Librería estándar (consola, ficheros, red, etc...)
- Utilidades (intérprete interactivo, depurador, paquetes)

Node.js

También puede verse como un entorno huésped para JS en el servidor

Node.js: orígenes



Tiene su origen en un proyecto de **Ryan Dahl** y sus colaboradores en la empresa *Joyent*, que fue presentado en una conferencia en la *JSConf* de 2009.

<https://youtu.be/ztspvPYyblY>

Escrito en C/C++ y JS

Objetivo del proyecto

→ Escribir aplicaciones muy eficientes en E/S con el lenguaje dinámico más rápido (v8) para soportar miles de conexiones simultáneas

Planteado sin complicaciones innecesarias

- Concurrencia sin paralelismo
- Lenguaje sencillo y muy extendido: JS
- API muy pequeña y muy consistente
- Apoyándose en Eventos y *Callbacks*

Paquetes: npm

Los módulos suelen distribuirse en forma de paquetes (*packages*) o aplicaciones:



- colección de módulos
- fichero *manifest* (JSON)
- fichero de ayuda (md)

describe el paquete y sus dependencias

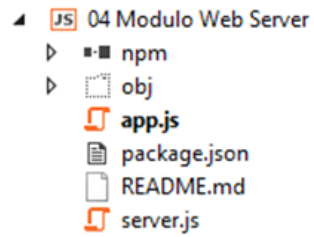


pueden publicarse de forma pública o privada



npm es la herramienta de instalación de paquetes incluida en Node.js

Los proyectos Node.js creados en Visual Studio ya responden a la estructura de paquetes, para facilitar la creación de estos



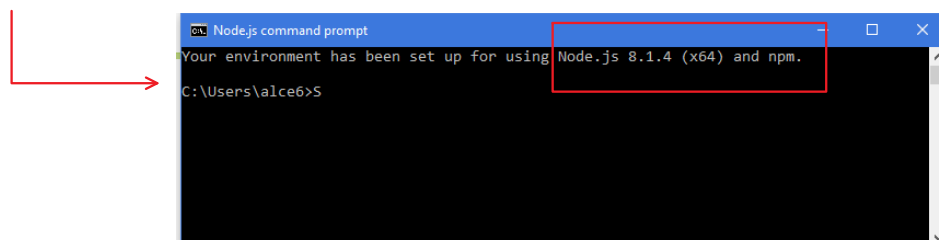
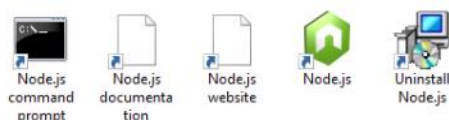
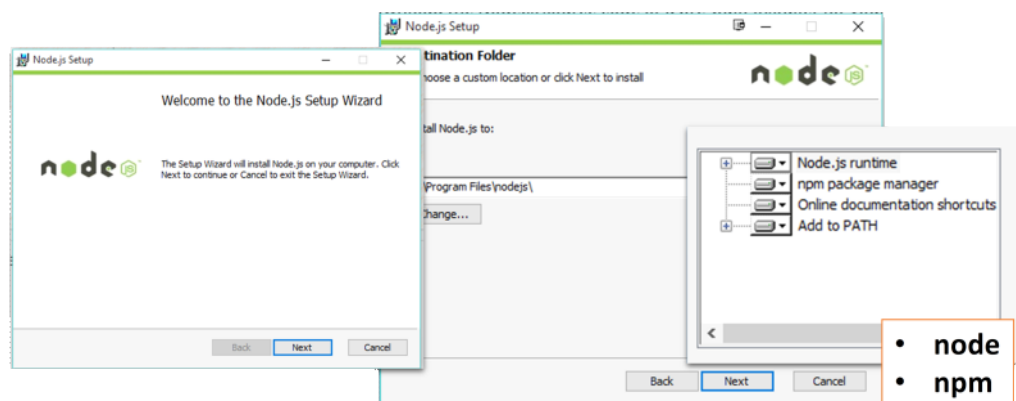
Instalación de Node.js & npm

sábado, 9 de septiembre de 2017 20:35

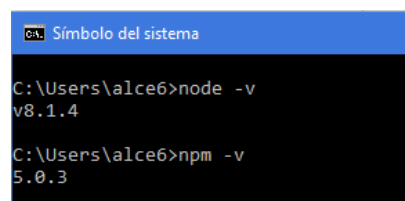
v8.4.0 Current

Latest Features

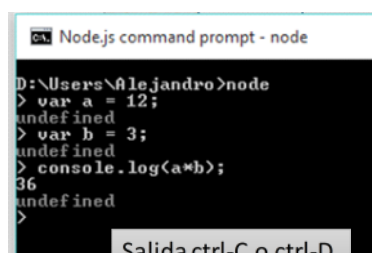
node-v8.4.0-x64.msi



- Accesible desde cualquier CLI (*cmd*, PowerShell...)
- Incluido en el *path*
- Se puede comprobar consultando la versión de *Node* y de *npm*



Con el comando "*node*" entramos en la consola de *Node*, un entorno REPL (*Read-Eval-Print-Loop*) similar a la consola de JS en los navegadores



Salida ctrl-C o ctrl-D

El mismo comando *node* <archivo.js>, permite la ejecución de un fichero

Uso de npm



<https://www.npmjs.com/>

*(Node)
package manager
for JavaScript*

```
C:\> npm install -g <package>
```

la g indica una instalación global, válida para todos los proyectos del usuario que realiza la instalación;
Por eso la carpeta empleada es

```
%users%\AppData\Roaming\npm\node_modules\<package>
```


Uso de npm: package.json



```
C:\> npm init
```

```
Press ^C at any time to quit.  
name: (Angular_Bootstrap) angular_bootstrap  
version: (1.0.0)  
description:  
entry point: (index.js) index.html  
test command:  
git repository:  
keywords: {  
  author: "angular_bootstrap",  
  license: (ISC) "version": "1.0.0",  
  About to write to D:\ "description": "Ejercicios del curso Bootstrap & Angular\r  
    "main": "index.html",  
    "scripts": {  
      "test": "echo \"Error: no test specified\" && exit 1"  
    },  
    "author": "",  
    "license": "ISC"  
  }  
}
```

```
C:\> npm install <package> --save
```

Instalación del entorno



```
npm install jquery --save
```

```
-- jquery@3.1.1
```

```
npm install bootstrap --save
```

```
-- bootstrap@3.3.7
```

```
npm install angular --save
```

```
-- angular@1.6.2
```

package.json

```
{ "name": "angular_bootstrap", "version": "1.0.0",  
  "description": "Ejercicios del curso Bootstrap & Angular\r  
Madrid, Febrero 2017\r Alejandro Cerezo Lasne",  
  "main": "index.html", "scripts": { "test": "echo \"Error:  
no test specified\" && exit 1" }, "author": "", "license":  
"ISC", "dependencies": { "angular": "^1.6.2",  
  "bootstrap": "^3.3.7", "jquery": "^3.1.1" }}
```


Scripts de npm

jueves, 29 de marzo de 2018 21:05

Podemos ejecutar comandos de un paquete mediante la definición de scripts en el archivo package.json

'npm start' no necesita especificar 'run', es el script estándar para lanzar un proyecto npm

```
"scripts": {
  "sass": "node-sass -o ./dist/ ./src/",
  "stylus": "stylus -w ./src/style.styl -o ./dist/style.css",
  "start": "supervisor main.js"
}

$ > npm run sass
$ > npm run stylus
$ > npm start
```



podemos reemplazar los gestores de tareas (*task runners*) por simples scripts NPM para automatizar tareas.

<https://www.keithcirkel.co.uk/how-to-use-npm-as-a-build-tool/>

Keith Cirkel

JavaScript Cyber Shepherd



09 Dec 2014 in Node.js, npm, JavaScript

How to Use npm as a Build Tool

```
package.json - boilerplate-npm - Visual Studio Code
Archivo Editar Selección Ver Ayuda

package.json
1 {
2   "name": "boilerplate-npm",
3   "version": "1.0.0",
4   "description": "Boilerplate de Automatización y Optimización de Tareas Front end con Scripts NPM",
5   "main": "package.json",
6   "scripts": {
7     "test": "echo \"Error: no test specified\" && exit 1",
8     "build:package": "rm -rf node-sass node-sass-cli babel-cli babel-preset-latest pug-cli browser-sync parallelshell onchange uncss postcss-cli autoprefixer buildify useref useref-cli html-minifier copy imagemin imagemin-cil imagemin-pngquant imagemin-jpeg-recompress",
9     "dev:package": "rm -rf node-sass node-sass-cli babel-cli babel-preset-latest pug-cli browser-sync parallelshell onchange uncss postcss-cli autoprefixer buildify useref useref-cli html-minifier copy imagemin imagemin-cil imagemin-pngquant imagemin-jpeg-recompress",
10    "sass": "node-sass -o ./src/css/ ./src/scss/",
11    "babel": "babel -w ./src/js/ -d ./src/js/",
12    "pug": "pug -w -P ./src/pug/ -o ./src/",
13    "dev:serve": "browser-sync start --server --directory --startPath ./src --serveStatic ./src --files ./src",
14    "dev:proxy": "browser-sync start --proxy localhost --directory --startPath ./classes/taller-fronted/boilerplate-npm/src --serveStatic / --files ./src",
15    "watch:metatag": "parallelshell 'npm run sass' 'npm run babel' 'npm run pug'",
16    "reload": "browser-sync reload",
17    "watch:reload": "onchange ./src -- npm run reload",
18    "watch:dev:serve": "parallelshell 'npm run watch:metatag' 'npm run dev:serve' 'npm run watch:reload'",
19    "watch:proxy": "parallelshell 'npm run watch:metatag' 'npm run dev:proxy' 'npm run watch:reload'",
20    "uncss:cli": "uncss http://bestian.com, http://bestian.com/cursos, http://bestian.com/acercas -o ./dist/css/styles.css",
21    "uncss:files": "uncss ./src/*.html -o ./dist/css/styles.css",
22    "autoprefixer": "postcss -o autoprefixer --autoprefixer.browsers ' > 5%, ie 10' -r ./dist/css/styles.css",
23    "min:css-js": "node buildify",
24    "useref": "useref -r ./src/**/*.html -o ./dist",
25    "htmlmin": "html-minifier --html5 --remove-comments --collapse-whitespace --input-dir ./dist/ --output-dir ./dist/",
26    "statify": "node statify",
27    "imagemin": "imagemin -r ./src/img/*.*(jpg|png) -o ./dist/img/ -p-pngquant -p-jpeg-recompress",
28    "webp": "imagemin -r ./dist/img/*.*(jpg|png) -o ./dist/img/ -p-webp",
29    "svgmin": "imagemin -r ./src/img/*.svg -o ./dist/img/ -p-svg",
30    "gifmin": "imagemin -r ./src/img/*.gif -o ./dist/img/ -p-gif",
31    "optimize": "mogrify -resize 1024x1024 -o ./dist/img/*.jpg",
32    "pngresize": "mogrify -resize 1024x1024 -o ./dist/img/*.png",
33    "clean": "rimraf ./dist/**",
34    "build:folders": "mkdirp ./dist/css ./dist/js ./dist/img",
35    "build:fronted": "rm -rf uncss:cli && npm run autoprefixer && npm run min:css-js && npm run useref && npm run htmlmin",
36    "build:media": "npm run statics && npm run imagemin && npm run webp && npm run svgmin && npm run gifmin",
37    "build:dist": "npm run clean && npm run build:folders && npm run build:fronted && npm run build:media",
38    "serve": "browser-sync start --server --directory --startPath ./dist --serveStatic ./dist --files ./dist",
39    "proxy": "browser-sync start --proxy localhost --directory --startPath ./classes/taller-fronted/boilerplate-npm/dist --serveStatic / --files ./dist",
40  },
41   "author": "Jonathan MirCha <jonmircha@gmail.com>",
42   "license": "MIT"
43 }
```

Identificación de las tareas

jueves, 29 de marzo de 2018 21:14

- Tareas de Compilación
 - Sass
 - Babel
 - Pug...
- Tareas en Tiempo Real
 - Servidor Web
 - Servidor Proxy
 - Recarga en vivo
 - Observar cambios
- Tareas de Publicación
 - CSS
 - Concatenación
 - Depuración
 - Prefijos
 - Minificación
 - JS
 - Concatenación
 - Minificación
 - Ofuscación
 - HTML
 - Reemplazar referencias
 - Minificación
 - Archivos
 - Archivos Estáticos
 - Limpieza de archivos
 - Creación de carpetas
 - Imágenes
 - Optimización
 - Compresión
 - Conversión
 - Redimensión



node-sass
babel-cli
babel-preset-latest
pug
pug-cli
browser-sync
parallelshell
rimraf
mkdirp
uncss
postcss-cli
autoprefixer
buildify
useref
useref-cli
html-minifier
copy
imagemin
imagemin-cli
imagemin-pngquant
imagemin-jpeg-recompress
imagemin-webp
imagemin-svgo
imagemin-gifsicle

Scaffolding

jueves, 29 de marzo de 2018 21:26

```
src
  es6
  img
  templates
  scss
  statics
dist
  css
  js
  img
```

Sass - scss

jueves, 29 de marzo de 2018 21:30

```
npm install -D node-sass babel-cli babel-preset-latest parallelshell
```

al mismo nivel que el package.json se crea .babelrc (se crea desde VSC)

```
{  
  "presets": [ "latest" ],  
  "plugins": []  
}
```

watch

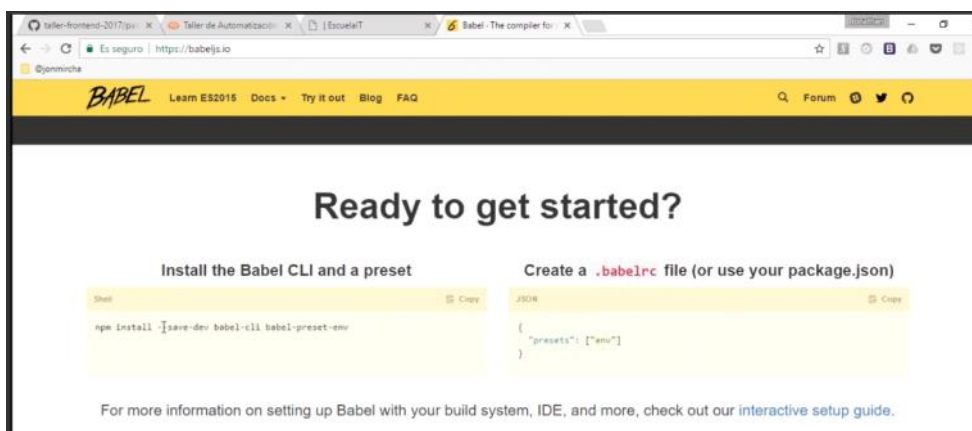
output

```
"sass": "node-sass -w -o ./dist/css ./src/scss",
```

```
"babel": "babel -w ./src/es6/ -d ./dist/js/",
```

watch

output



agrupación de las tareas

```
"watch:metalang": "parallelshell \"npm run sass\" \"npm run babel\"",
```

Empaquetado

sábado, 9 de septiembre de 2017 18:21

Construcción de proyectos / empaquetado

herramientas para procesar
los fuentes de la aplicación

- Reducción del tiempo de descarga
- Preprocesadores CSS
- Optimización del código, CSS, HTML
- Cumplimiento de estilos y
- Generación de JavaScript (transpilación)



<http://gruntjs.com/>



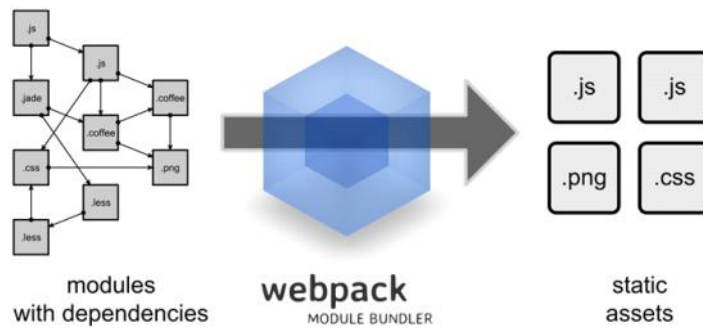
<http://gulpjs.com/>



<http://broccolijs.com/>



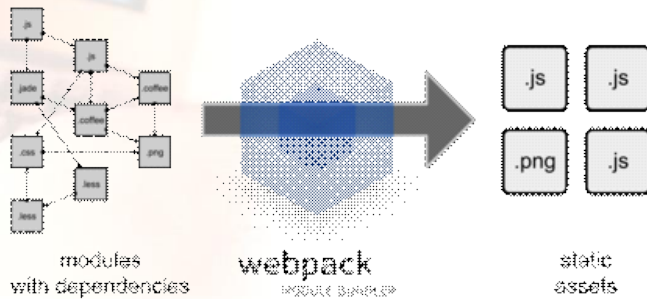
<https://webpack.github.io/>



finalmente incluido en **Angular-cli**

WebPack

martes, 27 de marzo de 2018 11:40



<https://webpack.js.org/>

webpack es una **herramienta de agregación** de recursos para aplicaciones web

permite generar una distribución única a partir de un conjunto establecido de *assets*.

Entre los recursos que es capaz de gestionar webpack, podemos encontrar

formatos soportados por defecto

- HTML
- JavaScript
- CSS,

formatos que necesitan ser transformados

- SASS
- TypeScript
- Jade

Con webpack definiremos un **único pipeline de procesamiento de recursos**,

obteniendo un **único fichero** empaquetado y minificado con todos los recursos necesarios para nuestro desarrollo

Documentación

<https://webpack.js.org/concepts/>

Sponsor webpack and get apparel from the official shop or get stickers here! All proceeds go to our open collective!

webpack

DOCUMENTATION | CONTRIBUTE | VOTE | BLOG | SEARCH | GITHUB | TWITTER | YOUTUBE

CONCEPTS | CONFIGURATION | API | GUIDES | LOADERS | PLUGINS

webpack v4.4.1

- Concepts
 - Entry
 - Output
 - Loaders
 - Plugins
 - Mode
- Entry Points
- Output
- Mode
- Loaders
- Plugins
- Configuration
- Modules
- Module Resolution
- Dependency Graph
- The Manifest
- Targets
- Hot Module Replacement

Concepts

At its core, webpack is a *static module bundler* for modern JavaScript applications. When webpack processes your application, it recursively builds a *dependency graph* that includes every module your application needs, then packages all of those modules into one or more *bundles*.

[Learn more about JavaScript modules and webpack modules here.](#)

Since v4.0.0 webpack does not require a configuration file. Nevertheless, it is *incredibly configurable*. To get started you only need to understand four **Core Concepts**:

- Entry
- Output
- Loaders
- Plugins

This document is intended to give a **high-level** overview of these concepts, while providing links to detailed concept specific use cases.

Entry

An **entry point** indicates which module webpack should use to begin building out its internal *dependency graph*. After entering the entry point, webpack will figure out which other modules and

Más información

<https://www.genbetadev.com/javascript/webpack-gestion-integrada-y-eficiente-de-tus-assets>

<https://carlosazaustre.es/primeros-pasos-con-webpack/>

Instalacion

martes, 27 de marzo de 2018

19:04

Prerrequisitos

- *NodeJS*
- *npm*

Instalación global

```
npm install webpack -g
```

Instalación específica en un proyecto

```
npm install webpack --save-dev  
npm install webpack-cli --save-dev
```

La instrucción que ejecutará webpack será

`node_modules/.bin/webpack`

Ejemplo básico

martes, 27 de marzo de 2018 19:08

Elementos del proyecto

index.html
app.js

```
<!DOCTYPE html>
<html lang="es">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-
scale=1.0">
  <meta http-equiv="X-UA-Compatible" content="ie=edge">
  <title>Uso de Webpack</title>
</head>
<body>
  <header><h1>Uso de Webpack</h1></header>
  <script src="bundle.js"></script>
</body>
</html>
```

referencia la bundle, resultante
del uso de webpack

script npm
(incluido en package.json)

"build": "webpack app.js -o bundle.js"

A diferencia de versiones anteriores, el fichero
de salida se indica mediante --output (-o)

El comando se ejecuta como

```
npm run build
```

```
> webpack app.js -o bundle.js
```

```
Hash: 487a20a73616cf0ffe96
Version: webpack 4.3.0
Time: 104ms
Built at: 2018-3-29 23:12:40
    Asset      Size  Chunks             Chunk Names
bundle.js  586 bytes      0  [emitted]  main
Entrypoint main = bundle.js
    [0] ./app.js 69 bytes {0} [built]
```

Fichero de configuración

jueves, 29 de marzo de 2018 23:15

webpack.config.js

```
const path = require('path');

module.exports = {
  entry: './app.js',
  output: {
    path: path.resolve(__dirname, 'dist'),
    filename: 'bundle.js'
  },
  mode: 'development'
};
```

```
"build": "webpack --config webpack.config.js"
```

app.js

```
import { addMensaje } from './module.js'
addMensaje()
```

module.js

```
export function addMensaje() {
  console.log("Ejemplo del uso de Webpack")
}
```

```
> webpack --config webpack.config.js
```

Hash: 9cae6e6282eb9287eb4a

Version: webpack 4.3.0

Time: 116ms

Built at: 2018-3-29 23:39:22

Asset	Size	Chunks	Chunk Names
bundle.js	3.57 KiB	main [emitted]	main
Entrypoint main = bundle.js			
./app.js	54 bytes	{main} [built]	
./module.js	82 bytes	{main} [built]	

Modo "escucha"

jueves, 29 de marzo de 2018

23:48

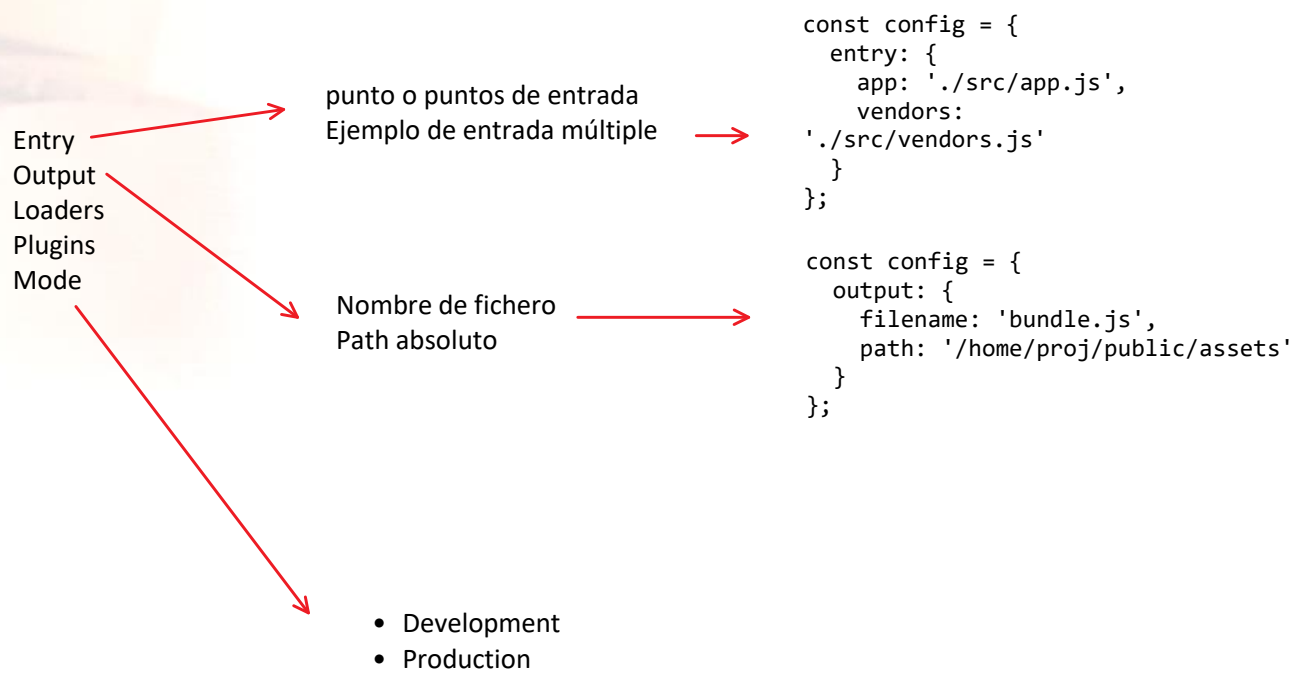
```
"scripts": {  
  "build": "webpack --watch --colors"  
}
```

Al correr `npm run build` tendremos ejecutando webpack en modo watch lo que significa que a cada cambio que hagamos en nuestro código, webpack creará el bundle de nuevo pero de una manera más rápida que por ejemplo con Browserify o Gulp ya que lo mantiene en memoria. Y con el flag `--colors` vemos de una manera más gráfica el resultado en la consola:

Elementos de la configuración

jueves, 29 de marzo de 2018

23:40



Loaders

jueves, 29 de marzo de 2018 23:45

Webpack permite que trabajemos con cualquier tipo de archivo (CSS, preprocesadores CSS, preprocesadores de JavaScript, imágenes, etc...) Con indicarle que *loader* debe utilizar e instalarlo, nos es suficiente.

```
module.exports = {
  module: {
    rules: [
      { test: /\.css$/, use: 'css-loader' },
      { test: /\.ts$/, use: 'ts-loader' }
    ]
  }
};
```

Cualquier recurso utilizado en la aplicación debe definirse y ser tratado como un módulo, independientemente de si es un fichero JavaScript, CSS o cualquier otro recurso.

Para que los estilos CSS sean minificados e incluidos en la distribución generada, tendremos que

- hacer un **require/import** de estos ficheros desde el punto de entrada principal (app.js) o
- incluirlos como "entry point"

Lo mismo pasará si queremos incluir alguna imagen, por ejemplo.

De esta manera, webpack los procesará y permitirá su aplicación y carga desde JavaScript directamente, evitando el tener referenciados los ficheros CSS mediante `link` o `style` dentro de la página HTML principal y minimizando así el número de peticiones que se hacen al servidor para completar la carga de la página.

Pre-procesadores de CSS : SASS

En el caso de sass, además de `style-loader` y `css-loader`, se necesitan `node-sass` y el `loader sass-loader`

```
npm install style-loader css-loader sass-loader node-sass --save-dev
```



Plugins

jueves, 29 de marzo de 2018 23:44

Junto con los *loaders*, que permiten incluir en el proceso diferentes tipos de archivos, existen *plugins* destinados a realizar cualquier otro tipo de tarea, como la optimización y mimificación en respuesta a diversas variables de entorno.

Los *plugins* son incluidos mediante un *require* y se instancian y se configuran en la opción *plugins*

Excluir el CSS del *bundle*

Especialmente en **producción**, mezclar dos tipos de outputs como son el procesamiento de JavaScript y el CSS, no es del todo deseable, ya que es importante favorecer el **cacheo de los recursos** en el navegador y si enviamos todos los recursos juntos, este fichero completo va a cambiar muchas más veces y se deberá descargar completo cada vez.

Para que los estilos se generen un fichero `.css` separado, tendremos que usar un plugin para webpack llamado **extract-text-webpack-plugin**. Este plugin permite detectar todas las definiciones de estilos, procesarlas y extraerlas del `bundle.js` general, permitiendo ser guardadas en el fichero que especifiquemos:

```
npm install -D extract-text-webpack-plugin@next
```

```
const ExtractTextPlugin = require("extract-text-webpack-plugin");

const extractSass = new ExtractTextPlugin({
  filename: "style.css",
  //disable: process.env.NODE_ENV === "development"
});

module.exports = {
  ...
  module: {
    rules: [{...}
  ]
},
plugins: [
  extractSass
];

use: extractSass.extract({
  use: [
    {loader: "css-loader"},
    {loader: "sass-loader"}
  ],
  // use style-loader in development
  fallback: "style-loader"
})
```

Se mantiene la importación del CSS desde JS, aunque luego se extraiga

```
app.js → import '../scss/style.scss'
```

Se añade la llamada al fichero CSS desde el HTML

```
<link rel="stylesheet" href="style.css">
```


Servidor de desarrollo

jueves, 29 de marzo de 2018 23:51

También tenemos la opción de crear un servidor web de desarrollo con webpack. Para ello debemos instalar otra dependencia que es webpack-dev-server:

```
npm install --save-dev webpack-dev-server
```

Y modificar nuestro fichero webpack.config.js con el siguiente bloque:

```
devServer: {  
  host: '0.0.0.0',  
  port: 8080,  
  inline: true  
},
```

Esto nos va a crear un servidor (basado en Node.js y Express) de desarrollo en local, en el puerto 8080 que servirá nuestra carpeta raíz. También se le puede indicar la carpeta inicial del servidor

```
contentBase: path.join(__dirname, "dist"),
```

Para iniciarlo creamos un nuevo script npm

```
"start" : "node_modules\\.bin\\webpack-dev-server",
```


Angular eject

viernes, 30 de marzo de 2018 12:15

<https://github.com/angular/angular-cli/wiki/eject>

Extrae la aplicación del contexto de `angular-cli` y en su lugar expone la configuración y los scripts de webpack

El comando `ng eject` tiene los mismos modificadores que `ng build` para generar la adecuada configuración inicial de web pack



- aot
- app
- base-href
- deploy-url
- environment
- extract-css
- force
- i18n-file
- i18n-format
- locale
- missing-translation
- output-hashing
- output-path
- poll
- progress
- sourcemap
- target
- vendor-chunk
- common-chunk
- verbose
- watch

Configuración de *proxies*

lunes, 7 de agosto de 2017 21:38

Configuración *git*

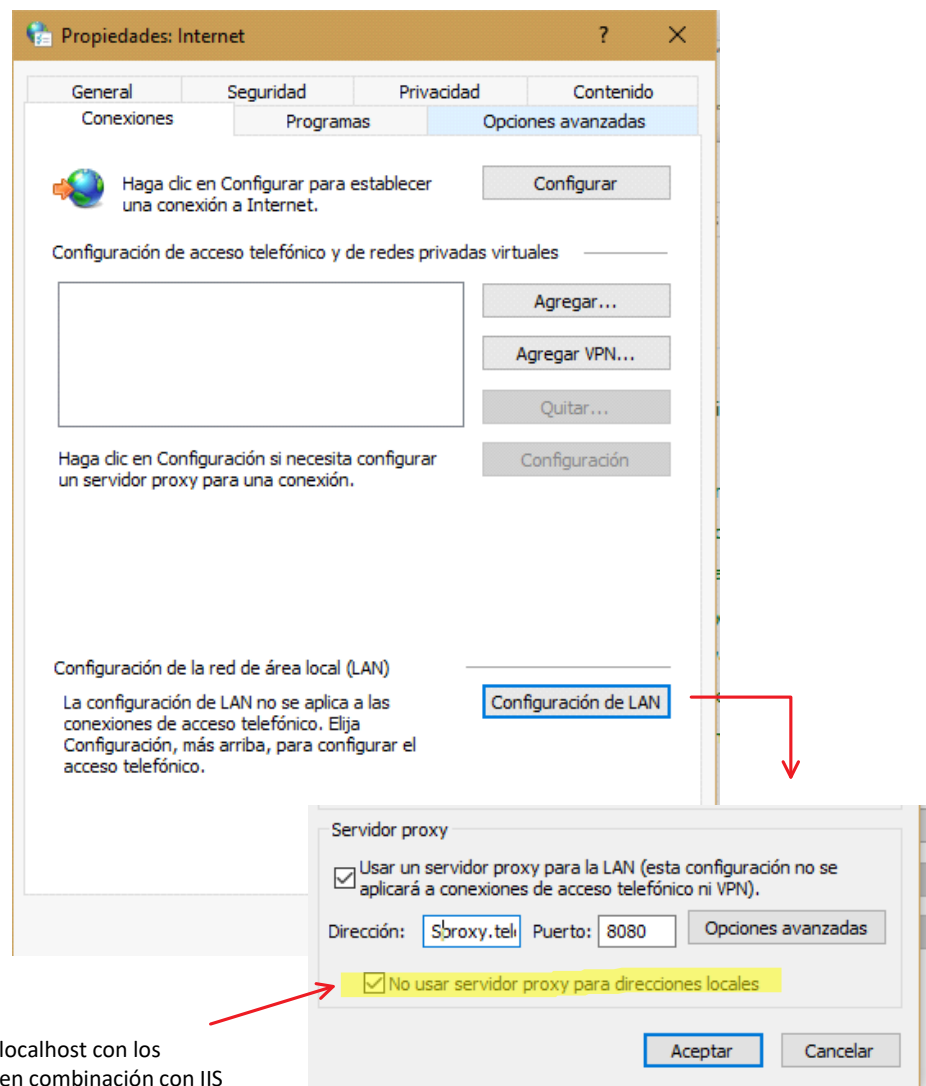
```
git config --global http.proxy http://user:passw@proxy.empresa.es:8080
```

Configuración npm

```
$>npm config set proxy http://user:passw@proxy.empresa.es:8080
```

```
$>npm config set https-proxy http://user:passw@proxy.empresa.es:8080
```

Propiedades de internet



Permite usar localhost con los navegadores en combinación con IIS para probar el código desarrollado