* **ANGULAR (PROGRAMACIÓN REACTIVA) / ANGULAR ROUTER**
* IONIC / CAPACITOR
* **NGRX** (EL REDUX DE ANGULAR)
* **RXJS** (PARA PROGRAMACION REACTIVA)
* RAMDA (PARA PROGRAMACION FUNCIONAL)
* GITFLOW: Para crear funciones que ejecuten acciones de GIT.

**TYPESCRIPT**

* Modularización
* Tipado fuerte
* Programación orientada a objetos

En REACT podemos usar FLOW en su lugar.

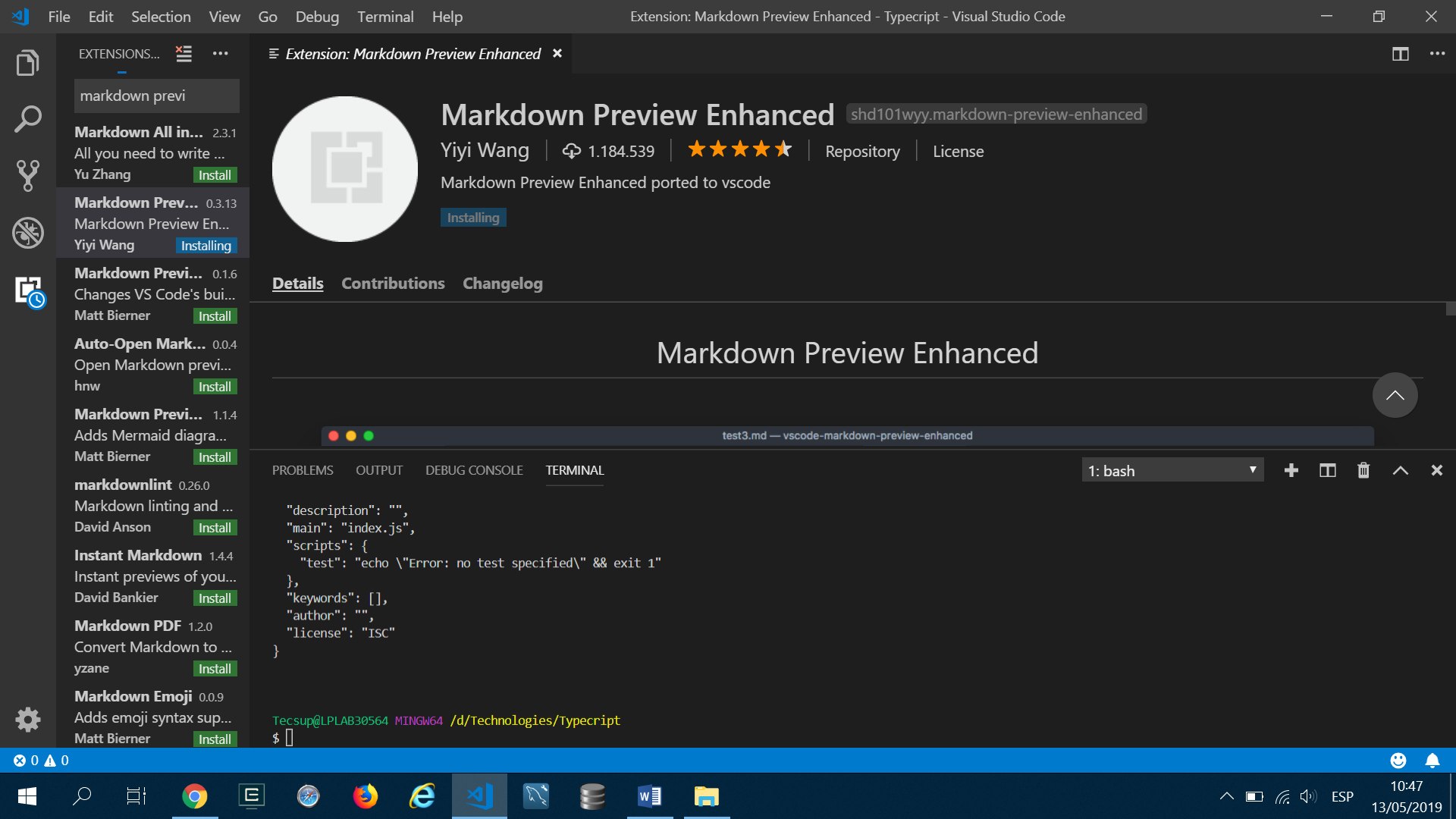
**MATERIAL UI:**se puede definir el theme para cada proyecto:

<https://material-ui.com/style/color/#color-tool>

**TERMS:**

 lazy evaluation?

**Para escribir en Markdown:**

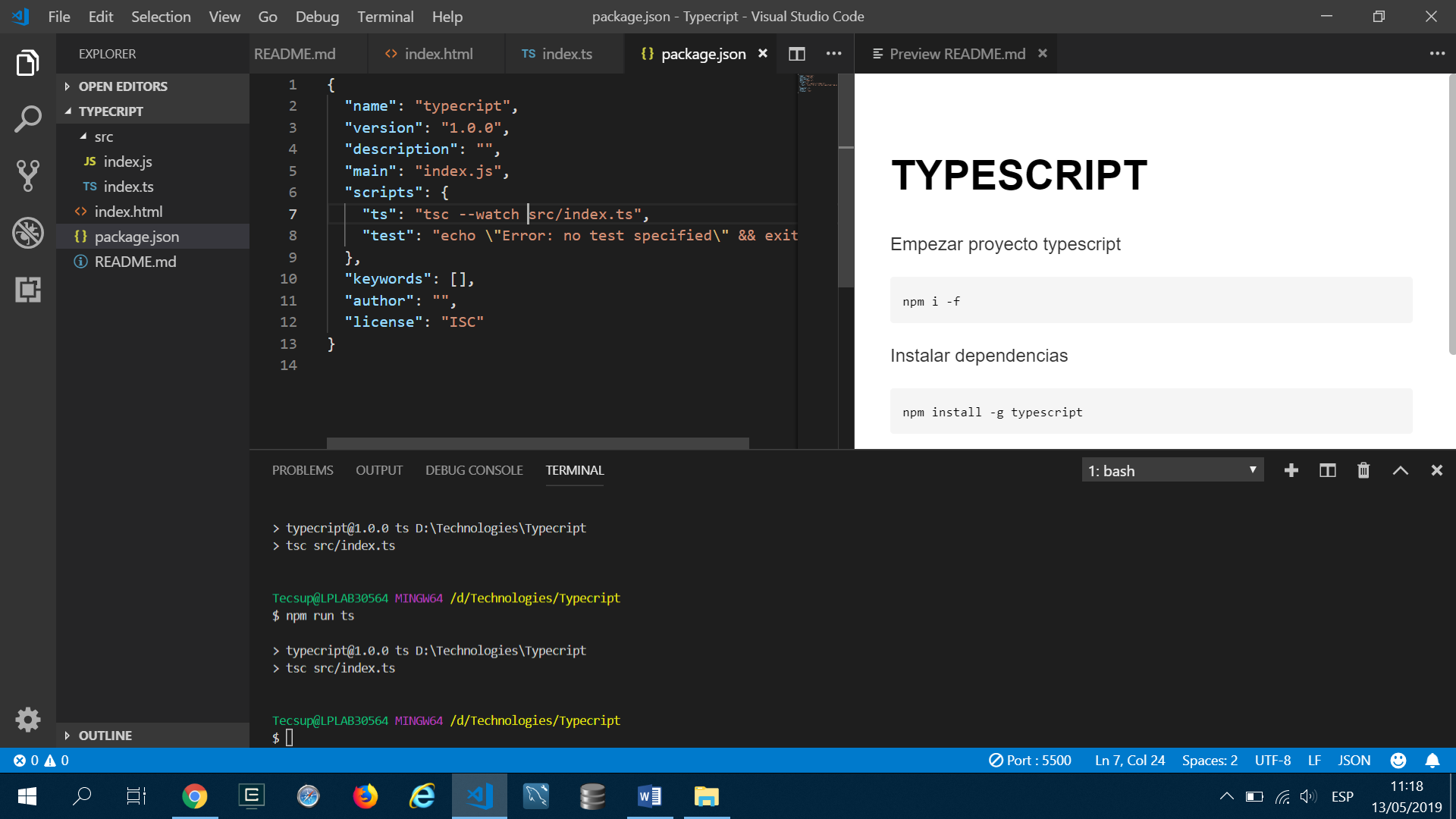


` ` `javascript

` ` `

GitHUB GIST(líneas de código embebidas para insertar en el README)

Para ver por cambios constantes:



Enum: custom types.

**INTERFACES: NEW CONCEPT**

**IMPLEMENTS:**

Se usa mucho en Angular

Ng: ciclos de vida en Angular.

Se usa implements en los ciclos de vida, es decir cada clase que herede esto debe tener si o si los parámetros definidos en el interface.

Public / Private / Protected

Private: Private stuff won’t be inherited.

Protected: Para que los hijos hereden algo que sea privado pero no public.

Agrupar clases: Namespace

Typescript con ReacT?

DECORATORS: Patrón de Javascript

Se extiende una clase, se ubica justo antes de la clase.

ANGULAR:

npm install -g @angular/cli

ejecutar ng para saber si se install.

* **Angular / NGXJ / RXJS (10 y 11)**
* React Native (12)
* VUE / Socket.io (13)
* PATRONES JS (14)
* Test Unitarios (JASMINE / KARMA / PROTRACTOR) (15)
* PWA + CI/CD/DOCKER (16)

AUTOMATIZACIÓN:

"jasmine-core": "~2.99.1",

"jasmine-spec-reporter": "~4.2.1",

"karma": "~4.0.0",

"karma-chrome-launcher": "~2.2.0",

"karma-coverage-istanbul-reporter": "~2.0.1",

"karma-jasmine": "~1.1.2",

"karma-jasmine-html-reporter": "^0.2.2",

"protractor": "~5.4.0",

Protractor: crea una especie de robot para monitorear la aplicación.

ANGULAR CONCEPTS:

Modules:

Recibe otros módulos y dentro de sus parámetros tiene “Bootstrap” Que es donde especificamos que inyectamos.

Components:

Estos tienen su html y css, tenemos template, estilos y un selector.

El selector es el nombre de la etiqueta que contendrá el componente.

Directives: Le agrega funcionalidades a un HTML

Services and Dependency Injection:  
provider: elemento que recibe otro código (dependency injection).