UNIVERSIDAD AUTONOMA DE AGUASCALIENTES

Materia: **Tecnologías Web**

Actividad

Cuestionario del ciclo de vida de un componente de angular y proyecto de ejemplo

Ingeniería Sistemas Computacionales Semestre 6° C

Javier Guerrero Carrera ID: 291590

Maestra: **GEORGINA SALAZAR PARTIDA**

Fecha de entrega: 28/03/23

El termino hooking ¿ a que está asociado?

Es una etapa de vida de un componente

2. ¿Qué es un hook en general y que es un hook en angular?

En programación, un hook (gancho en español) es un punto de entrada o salida en el flujo de ejecución de un programa que permite realizar acciones específicas en momentos determinados. En Angular, un hook es una función que se ejecuta en un momento específico del ciclo de vida de un componente y permite realizar tareas como inicialización, actualización o limpieza de recursos.

3. ¿Cuáles son los hooks ejecutados durante el ciclo de vida de un componente? Escríbelos,

no pegar imagen

ngOnInit(): se ejecuta después de que se ha inicializado el componente y se han resuelto las dependencias inyectadas.

ngOnChanges(changes: SimpleChanges): se ejecuta cuando cambia una propiedad vinculada al componente.

ngDoCheck(): se ejecuta durante cada detección de cambios y permite detectar cambios que no se detectan automáticamente.

ngOnDestroy(): se ejecuta justo antes de que se elimine el componente y permite realizar tareas de limpieza.

4. De la siguiente imagen ¿cuáles hooks pueden ejecutar las directivas y cuales las vistas –

contenido de los componentes?.

Los verdes son directivas y los azules los vista contenido

5. A cada hook le corresponde una interface ¿que encontramos en cual modulo?

A cada hook le corresponde una interface que se encuentra en el módulo @angular/core. Las interfaces correspondientes a los hooks son las siguientes:

ngOnInit

ngOnChanges

ngDoCheck

ngOnDestroy

6. ¿El lenguaje javascript maneja interfaces? Si o No y si tu respuesta es No ¿como lo

resuelve?

JavaScript no maneja interfaces de manera nativa, pero se pueden simular mediante el uso de objetos que implementan ciertas propiedades y métodos específicos. En TypeScript, que es un superset de JavaScript, se pueden definir interfaces de manera explícita y comprobar que un objeto las implementa correctamente en tiempo de compilación.

1. ¿Cuántas son las etapas por las que pasa un componente de angular?

Un componente de Angular pasa por ocho etapas en su ciclo de vida.

2. ¿Nombre que recibe en general cada etapa del ciclo de vida de un componente?

Cada etapa del ciclo de vida de un componente se llama hook.

3. ¿Por qué es necesario conocer el ciclo de vida de un componente? Además del link que te

deje para esta sección, en este otro link puedes leer una reflexión de porque necesitas

saber de esto https://blog.enriqueoriol.com/2018/10/ciclos-de-vida-en-angular-la-guiadefinitiva.html

Es necesario conocer el ciclo de vida de un componente para entender cómo funciona Angular, para optimizar el rendimiento de la aplicación y para solucionar problemas relacionados con el comportamiento del componente.

4. Un componente ¿es una clase de TypeScript? SI o NO

si

5. Con respecto al ciclo de vida de un componente ¿en que momento se ejecuta el

constructor de la clase?

El constructor de la clase se ejecuta antes de la primera etapa del ciclo de vida de un componente, que es ngOnChanges.

6. ¿Cuál es el mejor lugar para inyectar dependencias al componente?

El mejor lugar para inyectar dependencias en un componente es en el constructor de la clase.

7. Después de ejecutar el constructor ¿que hace Angular?

Después de ejecutar el constructor, Angular llama al método ngOnChanges, que es la primera etapa del ciclo de vida del componente.

8. ¿Cuántos métodos define cada interface utilizada en cada momento del ciclo de vida?

Cada interface utilizada en cada momento del ciclo de vida define varios métodos

9. ¿Como se escribe el nombre de una interfaz utilizada en los diferentes momentos del ciclo

de vida? Menciona un ejemplo

El nombre de una interfaz utilizada en los diferentes momentos del ciclo de vida comienza con "On", seguido del nombre de la etapa en la que se utiliza la interfaz, por ejemplo:

OnChanges

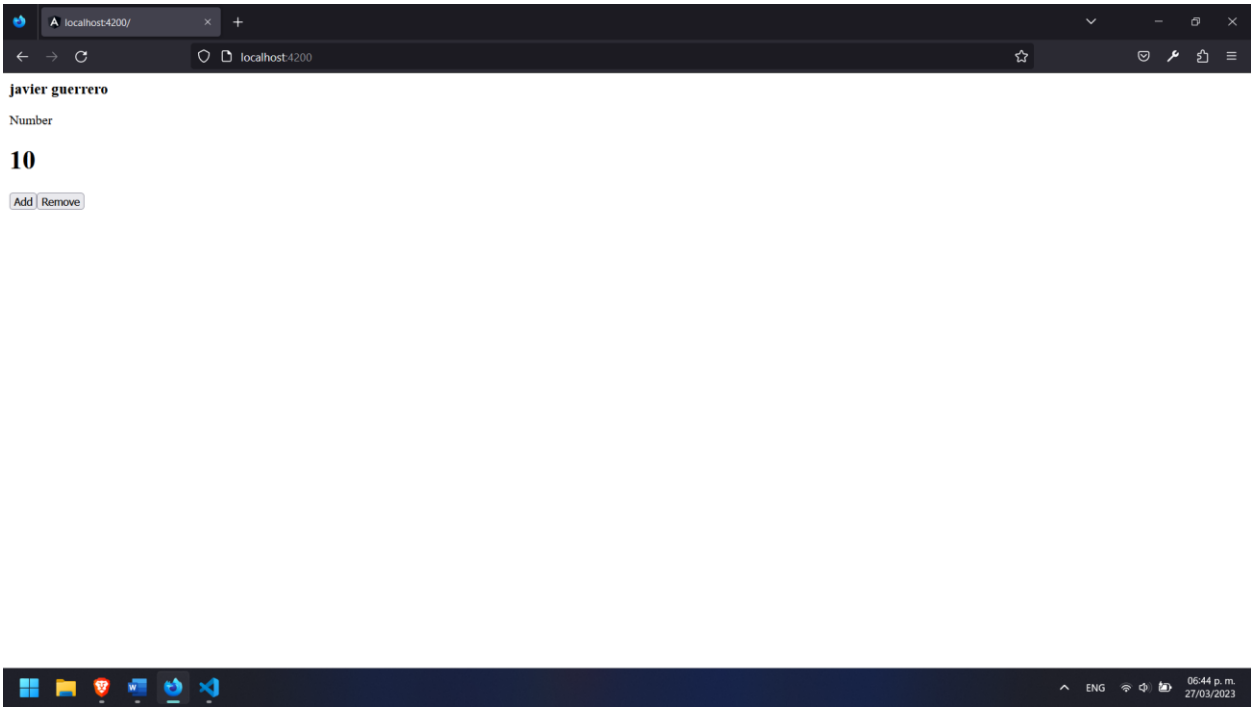
OnInit

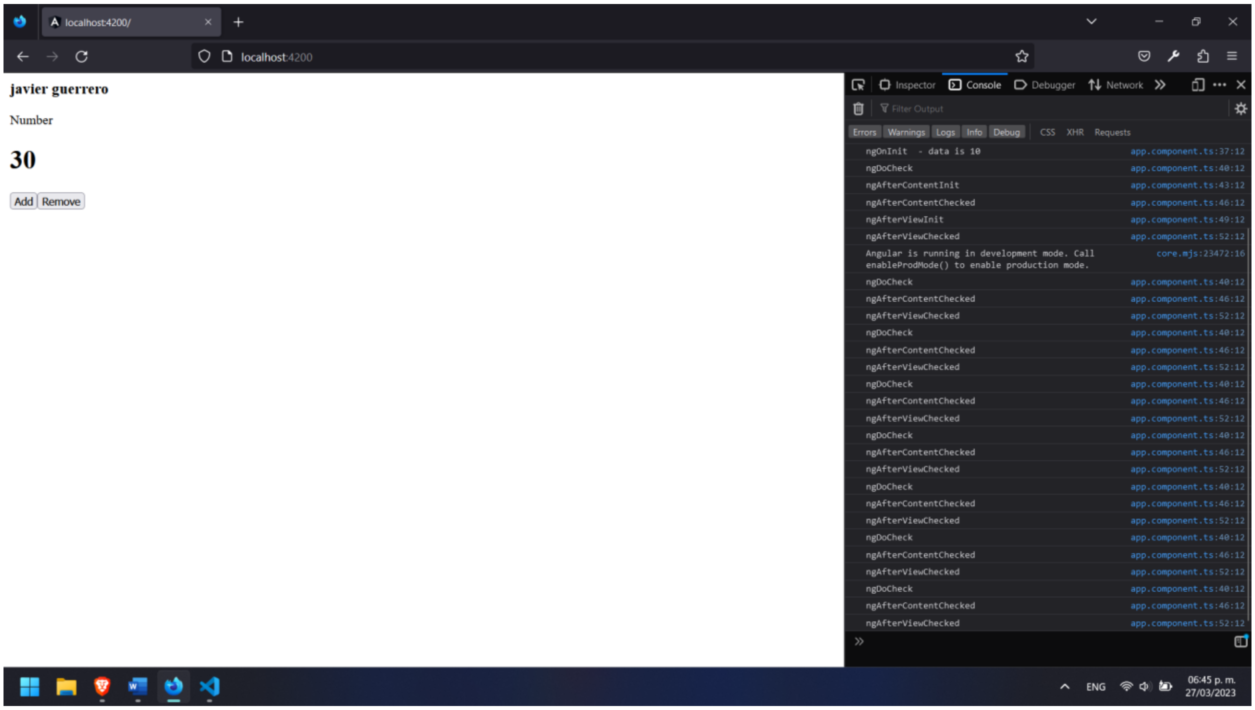
OnDestroy

10. Completar con texto en otro color lo que va en cada linea ….. Es importante saber que el compilador de TypeScript basado en navegador __no genera un error de compilación_ cuando _no__ implementamos funciones de interfaz en nuestra clase. Pero, en el __tiempo de compilación_ del código TypeScript, __arrojará un error.___

Hook	Descripción	Momento de ejecución
constructor	Este es el primer método que se ejecuta cuando se crea una instancia de un componente. Se utiliza para inicializar variables y configuraciones iniciales del componente.	Cuando se crea la instancia del componente.
ngOnInit	Este método se llama justo después del constructor y es el lugar ideal para hacer configuraciones adicionales. Por ejemplo, obtener datos de una API.	Después de la inicialización del constructor.
ngOnChanges	Se llama cada vez que hay un cambio en las propiedades de entrada de un componente. Puede ser útil para actualizar valores y hacer cosas adicionales en función de los cambios de entrada.	Cuando se detectan cambios en las propiedades de entrada de un componente.
ngDoCheck	Este método se ejecuta cada vez que se realiza una verificación de cambios. Puede ser útil para detectar cambios que no están cubiertos por ngOnChanges.	Después de cada verificación de cambios.
ngOnDestroy	Este método se llama justo antes de que se destruya un componente. Se utiliza para limpiar los recursos y evitar posibles fugas de memoria.	Antes de la destrucción del componente.
ngAfterViewInit	Se llama después de que la vista del componente ha sido inicializada. Esto significa que los elementos del DOM ya están disponibles y se pueden manipular.	Después de la inicialización de la vista del componente.

ngAfterContentInit	Este método se llama después de que el contenido proyectado se haya inicializado. Esto significa que los elementos que se proyectan en el componente están disponibles y se pueden manipular.	Después de la inicialización del contenido proyectado del componente.
ngAfterViewChecked	Este método se llama después de cada verificación de cambios y se utiliza para manipular el DOM.	Después de cada verificación de cambios.
ngAfterContentChecked	Se llama después de que el contenido proyectado haya sido verificado. Se utiliza para manipular el DOM de los elementos proyectados.	Después de cada verificación de cambios del contenido proyectado.




localhost:4200/
localhost:4200
javier guerrero
Number
30
Add
Remove
Inspector
Console
Debugger
Network
Filter Output
Errors
Warnings
Logs
Info
Debug
CSS
XHR
Requests
ngOnInit - data is 10
app.component.ts:37:12
ngDoCheck
app.component.ts:40:12
ngAfterContentInit
app.component.ts:43:12
ngAfterContentChecked
app.component.ts:46:12
ngAfterViewInit
app.component.ts:49:12
ngAfterViewChecked
app.component.ts:52:12
Angular is running in development mode. Call enableProdMode() to enable production mode.
core.mjs:23472:16
ngDoCheck
app.component.ts:40:12
ngAfterContentChecked
app.component.ts:46:12
ngAfterViewChecked
app.component.ts:52:12
ENG
06:45 p. m.
27/03/2023