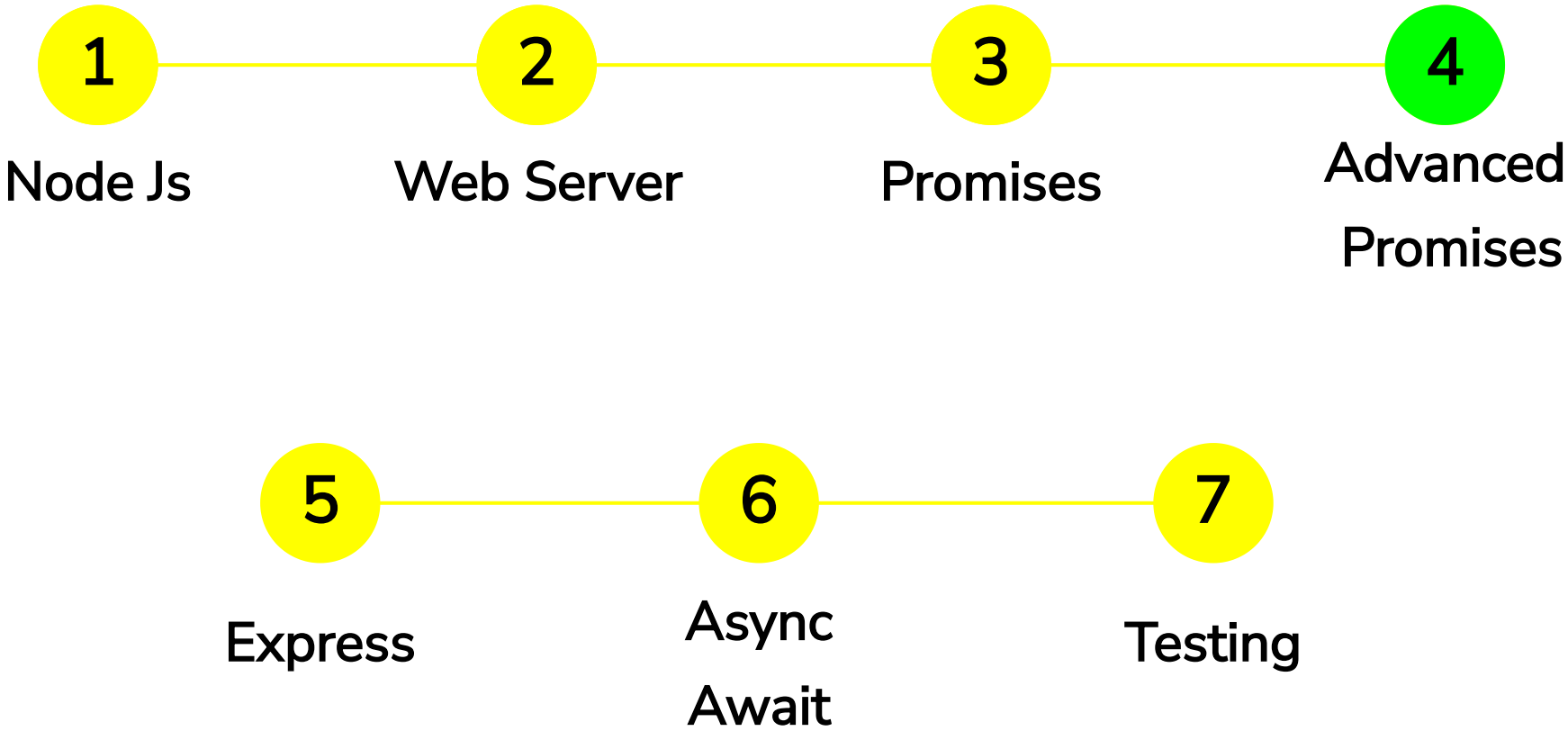

HENRY

Lecture

Advanced Promises

Contenido M3



Recapitulemos...

1. Promesa -> *Es un objeto que representa el resultado de una operación asincrónica que se resuelve con un valor o se rechaza con una razón.*

2. Status -> *Es una de las propiedades del objeto Promesa, la cual puede tener los estados pending, fulfilled o rejected.*

3. Pending -> *Es el estado inicial de la promesa antes de tomar uno de los dos caminos: fulfilled o rejected.*

4. Fulfilled -> *Es la promesa completada, significa que todo salió OK.*

Recapitulemos...

5 Rejected -> *Es la promesa rechazada o fallida.*

6 Value -> *Si la promesa es fulfilled nos retorna un valor, si es rejected nos retorna una razón.*

7 .then() -> *Es un método que devuelve una promesa en donde su valor de resolución es el resultado de los handlers: la data o la razón del rechazo.*

8 .catch() -> *Es un método que devuelve una promesa en donde su valor de resolución es la ejecución del errorHandler cuando la promesa es rechazada.*

OBJETIVO DE LA CLASE

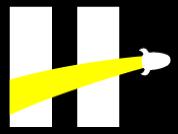
Profundizar en las promesas, conocer el comportamiento y encadenamiento del .then() e identificar el flow que tienen las promesas.

¿Qué veremos hoy?

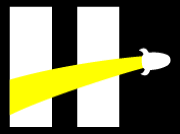
- Promesas
- .then()
- Flow de las promesas

Espacio de interacción

Recuerden hacer las preguntas en el **Q&A** con contexto para que no se pierdan en el chat.



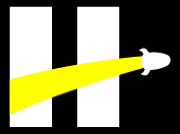
Promesas



Promesas

.then retorna una nueva promesa!

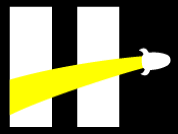
```
1
2 promiseB = promiseA.then(successHanlder, FailureHandler);
```



Promesas

Por esto podemos encadenarlas!

```
1
2  promiseA
3          .then(hacerAlgo)
4          .then(hacerAlgoMas)
5          .then(masCosas)
6          .catch(errorHandler);
7
8
9  //
10 .catch(errorHandler)
11
12 // es equivalente a
13
14 .then(null, errorHandler)
```



Promesas

FlowChart

Resumen

- **Promise:** Es un objeto de clase que representa la eventual finalización o rechazo de una operación asíncrona.

- **Status:** Una promesa puede estar en uno de estos estados:
  - Pending: Estado inicial, ni cumplida, ni rechazada.
  - fulfilled: Significa que la operación se completó con éxito.
  - rejected: Significa que la operación falló.

- **Handlers:** Son los manejadores o controladores que cuando una promesa se cumple con un valor o se rechaza con una razón, se ejecutan el controlador asociado mediante el .then() de una promesa.

Resumen

- **Handlers**:
  - successHandler: Si la promesa se cumple se llama este handler, el cual su valor de resolución es la información del resultado de la promesa.
  - errorHandler: Si la promesa falla se ejecuta este handler, el cual su valor de resolución es la razón del rechazo.

- **.then():** Devuelve una nueva promesa, diferente a la original.

- **Encadenamiento:** Cuando dos o más operaciones asíncronas consecutivas deben resolverse o rechazarse se pueden encadenar con el .then(), donde cada promesa representa la terminación del paso anterior asíncrono o síncrono en el .then()

Resumen

- **.catch()**: Es un método que devuelve una promesa en donde su valor de resolución es la ejecución del errorHandler cuando la promesa es rechazada.

- **Crear promesas:** Para definirla llamamos la clase Promise y la instanciamos, esta clase recibe un callback con los argumentos resolve y rejected.

¿Preguntas?

Homework

Pair Programming

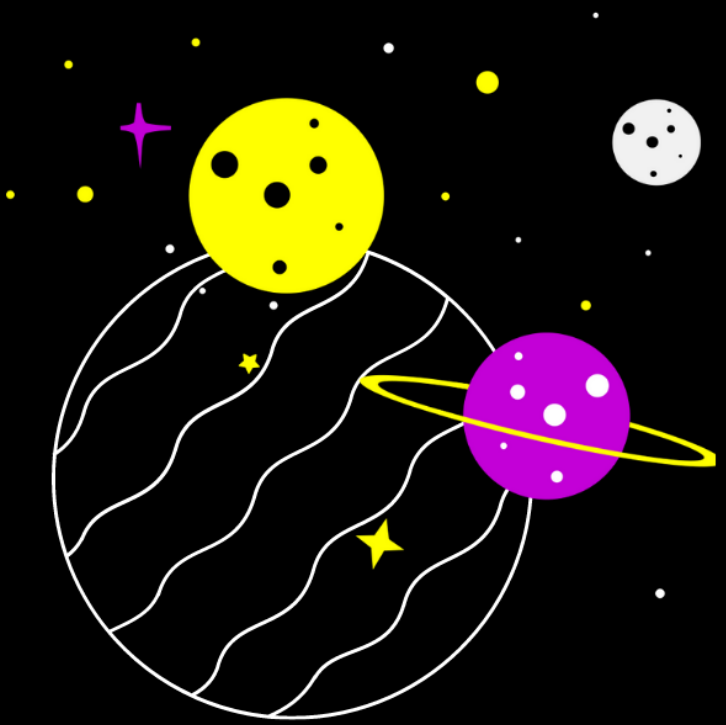
SUP

¿Preguntas?

PROXIMA CLASE

Express

HENRY



¡Muchas gracias!