Unlocking the Power of **Pure Functions** in JavaScript

SWIPE TO LEARN

RAJ BHENSDADIYA

- Pure functions are the **unsung heroes** of JavaScript.

- They are **essential concepts** that every developer should understand.

- Let's dive into **what** pure functions are and **why** they're so crucial!

What Are Pure Functions?

- Pure functions are **like mathematical equations** - they **always produce the same** result for the same input.

- And they have **two key characteristics**:

What Are Pure Functions?

- **Deterministic:** Their output solely depends on their input.
- **No Side Effects**: They don't modify external state.

What Are Pure Functions?

JS pureFunctions.js

```
function calculateTax(amount, taxRate) {
  return amount * (1 + taxRate);
}
```

RAJ BHENSDADIYA

What Are Pure Functions?

- In the real world, think of an e-commerce system.
- **Calculating the total cost** (input) including taxes always **gives you the same result** (output) regardless of the context.

RAJ BHENSDADIYA

Why "Pure"?

- Pure functions are called "pure" because they're **untainted by side effects** - they **don't interact** with the **external world**, making them predictable and reliable.

Why "Pure"?

JS impureFunc.js

```
let userCount = 0;
function registerUser(userInfo) {
  userCount++;
  // Modifies external state - side effect!
  saveToDatabase(userInfo);
}
```

Why "Pure"?

- In contrast, impure functions like this one not only create a new user but also alter the global userCount and save data to a database (side effects), making them harder to reason about.

Benefits of Pure Functions

- Why do we love pure functions?

- Because they make **our code cleaner**, **more testable**, and **reliable**!

- Here's why:

Benefits of Pure Functions

- **Predictable:** Always return the same result for the same input.
- **Testable:** Perfect for unit tests.
- **Concurrency:** Safe for parallel execution.

Benefits of Pure Functions

JS pureFunc.js

```
function findUniqueElements(arr) {
  return [...new Set(arr)];
}
```

RAJ BHENSDADIYA

Benefits of Pure Functions

- Just like a coffee machine that consistently brews your favorite cup (input) into the same delicious brew (output), pure functions ensure consistent results!

RAJ BHENSDADIYA

Impure Functions: The Trouble Makers

- Impure functions are the **rebels in the JavaScript** world.
- They **break the rules** by **causing side effects**, making code harder to understand and maintain.

Impure Functions: The Trouble Makers

JS impureFunc.js

```
function updateStock(product, quantity) {
  product.stock -= quantity;
  // Modifies external state - side effect!
  updateInventoryDatabase(product);
}
```

RAJ BHENSDADIYA

Impure Functions: The Trouble Makers

- Imagine an inventory system where updating stock (input) not only modifies the product but also updates a remote database (side effect) - impure functions can lead to unpredictable outcomes.

RAJ BHENSDADIYA

Dealing with Impure Functions

- Handling impure functions is like dealing with volatile chemicals.

- We should:
- **Isolate Side Effects:** Keep impurity contained.
- **Document Clearly:** Explicitly state and document side effects.
- **Testing**: Rigorously test impure functions.

RAJ BHENSDADIYA

Dealing with Impure Functions

JS isolateImpureFunc.js

```
function updateStock(product, quantity) {
  const updatedProduct = { ...product, stock: product.stock - quantity };
  updateInventoryDatabase(updatedProduct); // Isolate side effect
}
```

RAJ BHENSDADIYA

Dealing with Impure Functions

- By isolating the side effect (database update) and making it explicit, impure functions become more manageable, like working with controlled experiments in a lab.

I'm Raj.

Software Engineer;
passionate for building and shipping scalable software.

I am open for remote based contract job in React.js/ MERN/ Full-stack or Software Engineering.

Let's connect and expand our professional network together.

I'm excited to collaborate, exchange ideas, and contribute to impactful projects.