OpenAI

Prototype vs constructor count

Let's analyze this line by line 👇

🧩 Code:

```
js

function Counter() {}
Counter.prototype.count = 0;

console.log("CounterPrototype--->>", Counter.count)
```

🔍 Step-by-step Explanation:

1. **`function Counter() {}`**
   This creates a constructor function named **`Counter`**.

2. **`Counter.prototype.count = 0;`**
   - Here, you are adding a property called **`count`** to the **prototype object** of **`Counter`**.
   - So internally, it looks like this:

     ```
     js

     Counter.prototype = { count: 0, constructor: Counter }
     ```

   - Any instance created with **`new Counter()`** will inherit this **`count`** property from the prototype.

3. **`console.log("CounterPrototype--->>", Counter.count)`**
   - Here, you are accessing **`count`** **on the function object itself**, i.e., the constructor function **`Counter`**, **not on its prototype**.
   - But you never defined a property **`count`** directly on **`Counter`** — only on **`Counter.prototype`**.

   ✅ **`Counter.prototype.count`** exists
   ❌ **`Counter.count`** does **not** exist

🧠 Output:

```
bash

CounterPrototype--->> undefined
```

💡 Key takeaway:

- **`Counter.prototype.count`** → refers to a property on the prototype (used by instances)
- **`Counter.count`** → refers to a property on the constructor function itself (not related to prototype)

If you log the prototype’s property:

```js
console.log("CounterPrototype--->>", Counter.prototype.count)
```

👉 Output will be:

```lua
CounterPrototype--->> 0
```