Title: What are pure functions?

Answer:

The whole concept of a pure function is consistency and predictability (which IMO are keys to writing great software). The reason for the consistency and predictability is because pure functions have the following characteristics.

- Pure functions always return the same result given the same arguments.
- Pure function's execution doesn't depend on the state of the application.
- Pure functions don't modify the variables outside of their scope.

If a function passes *all three* of these requirements, it’s a pure function. If it fails even *one* of these, then it’s an *impure* function.

When you call a function that is "pure", you can predict exactly what's going to happen based on its input. This makes functions that are pure easy to reason about and testable.

Let's look at some examples.

```
function add (x,y) {
  return x + y
}
```

Though simple, add is a pure function. There are no side effects. It will always give us the same result given the same arguments.

Let's now look at two native JavaScript methods. .slice and .splice

```
var friends = ['Mikenzi', 'Jordyn', 'Merrick']
friends.slice(0, 1) // 'Mikenzi'
friends.slice(0, 1) // 'Mikenzi'
friends.slice(0, 1) // 'Mikenzi'
```

Notice .slice is also a pure function. Given the same arguments, it will always return the same value. It's predictable.

Let's compare this to .slice's friend, .splice

```
var friends = ['Mikenzi', 'Jordyn', 'Merrick']
friends.splice(0, 1) // ["Mikenzi"]
friends.splice(0, 1) // ["Jordyn"]
friends.splice(0, 1) // ["Merrick"]
```

.splice is not a pure function since each time we invoke it passing in the same arguments, we get a different result. It's also modifying state.

Tags: functions / methods