Crack Your JavaScript Interview Like a Pro! 🚀

I recently interviewed at multiple top tech companies, including **Microsoft, Amazon, Salesforce, Adobe, and PayPal.** These JavaScript questions were frequently asked in those interviews. If you're preparing for FAANG and other top-tier companies, mastering these will give you an edge!

by Gourav Roy

Difference between var, let, and const?

- var: Function-scoped, can be re-declared and updated.
- let: Block-scoped, can be updated but not re-declared.
- const: Block-scoped, cannot be updated or re-declared.

What is hoisting in JavaScript ?

- JavaScript moves function and variable declarations to the top of their scope before execution.
- `var` is hoisted with `undefined`, `let` and `const` are hoisted but not initialized.

Explain closures with an example.

- A closure is a function that remembers the variables from its outer scope even after the outer function has executed.

```
function outer() {
  let count = 0;
  return function inner() {
    count++;
    console.log(count);
  };
}
const counter = outer();
counter(); // 1
counter(); // 2
```

What are arrow functions, and how do they differ from regular functions?

- Shorter syntax, do not have their own this, cannot be used as constructors.

```
const add = (a, b) => a + b;
```

Explain this keyword in JavaScript.

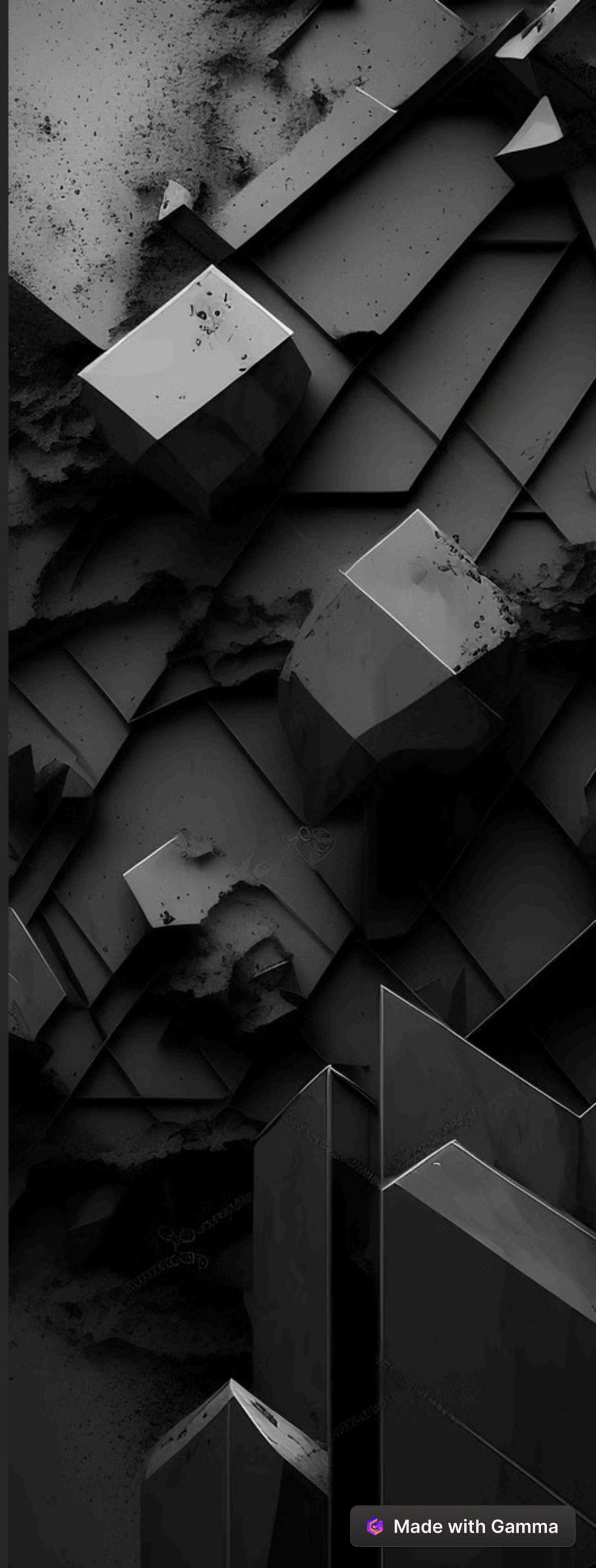
- Refers to the object that is executing the function.
- this in global scope refers to window (browser) or global (Node.js).
- In arrow functions, this is inherited from the surrounding scope.

What is the difference between synchronous and asynchronous code?

- Synchronous: Code runs sequentially, blocking further execution.
- Asynchronous: Code runs in the background without blocking execution.

Explain Promises and their states.

- A Promise represents an asynchronous operation with three states:
  - Pending: Initial state.
  - Fulfilled: Operation successful.
  - Rejected: Operation failed.



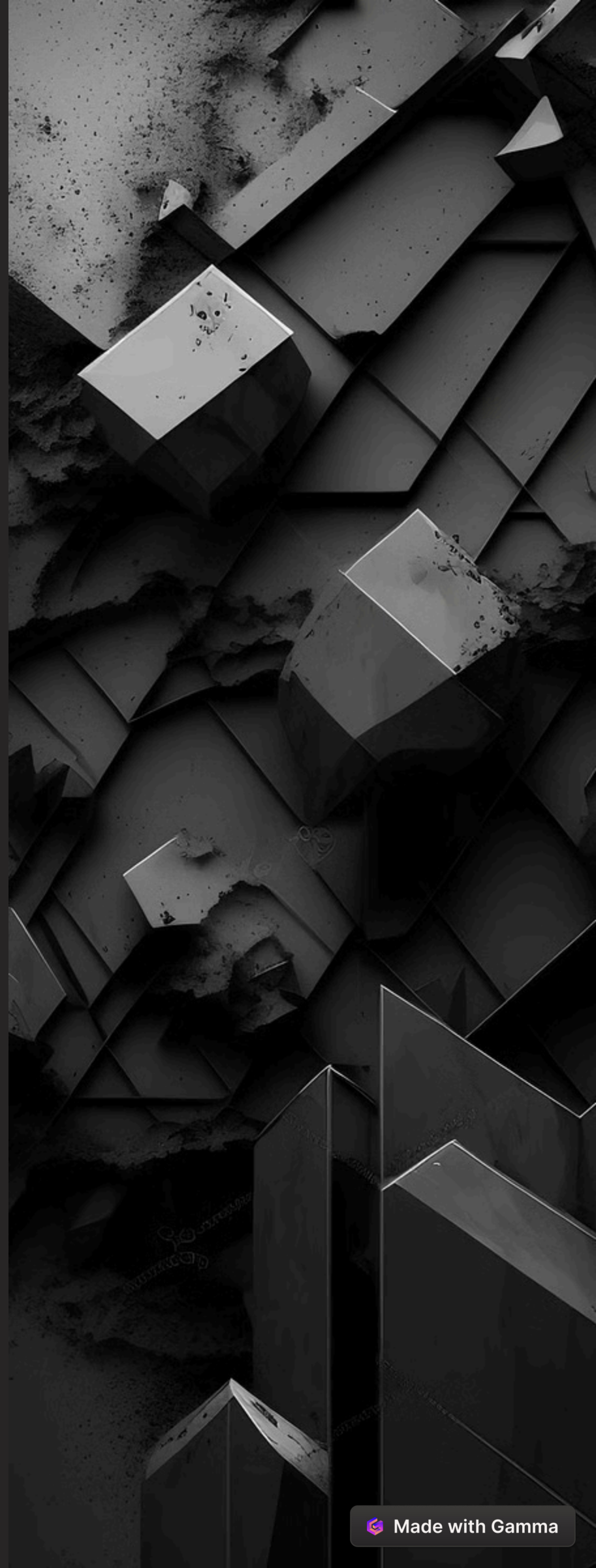
What is async/await, and how is it different from Promises?

- async makes a function return a Promise.
- await pauses execution until the Promise resolves.

```
async function fetchData() {
  let data = await fetch(url);
  console.log(await data.json());
}
```

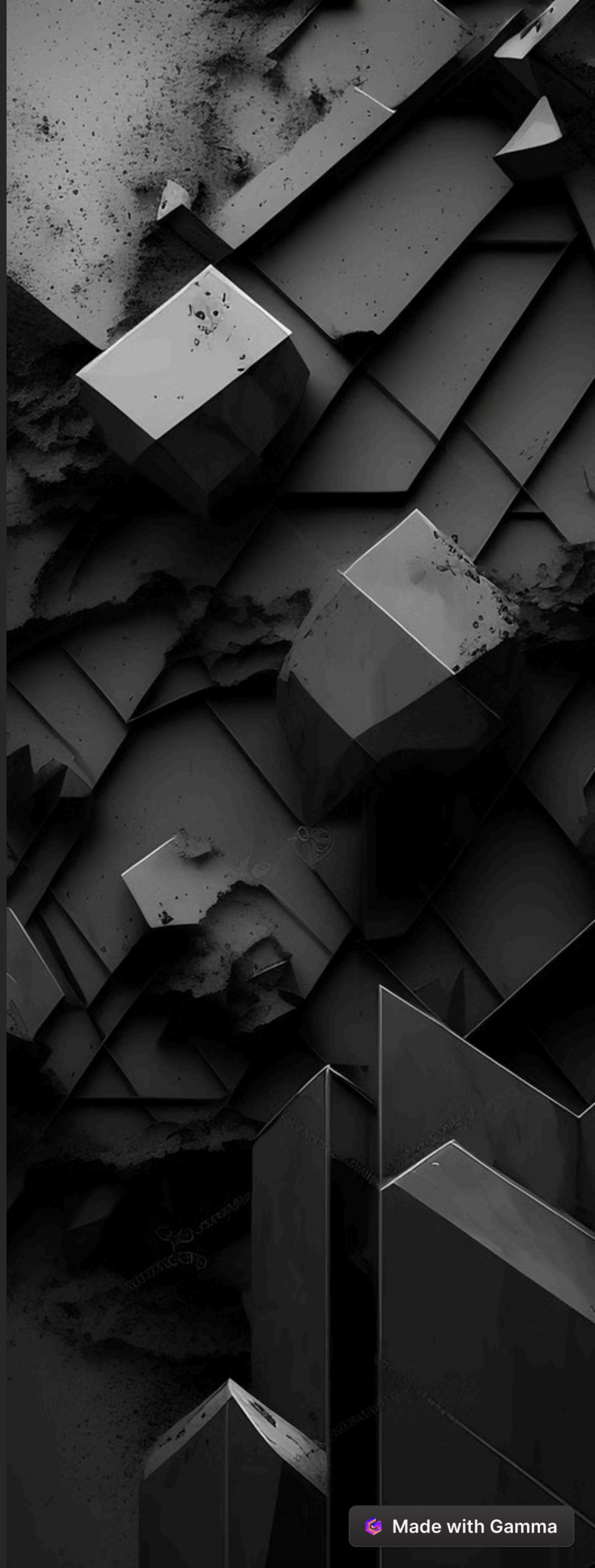
What is the event loop in JavaScript?

- Handles asynchronous operations by executing tasks from the callback queue after the main thread is free.



What are JavaScript prototypes and prototype chaining?

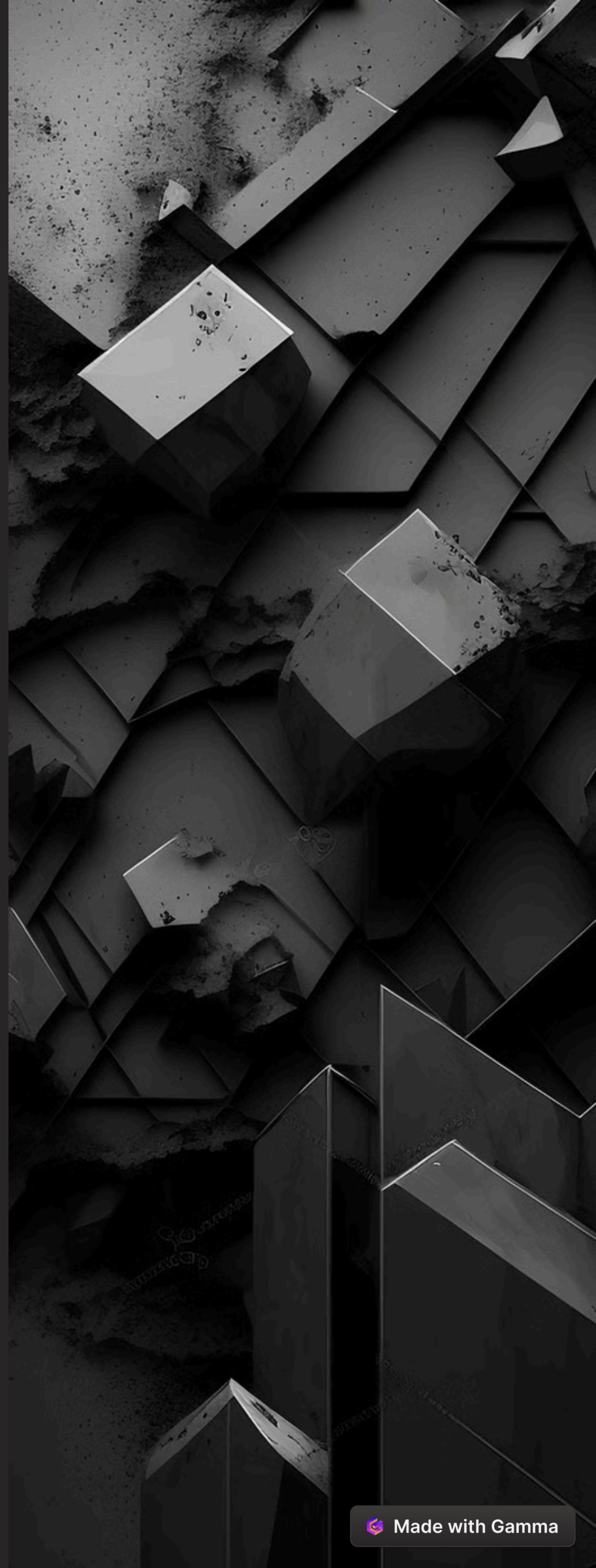
- Objects inherit properties/methods from a prototype.
- Prototype chaining allows objects to inherit from other objects.



What are higher-order functions? Give an example.

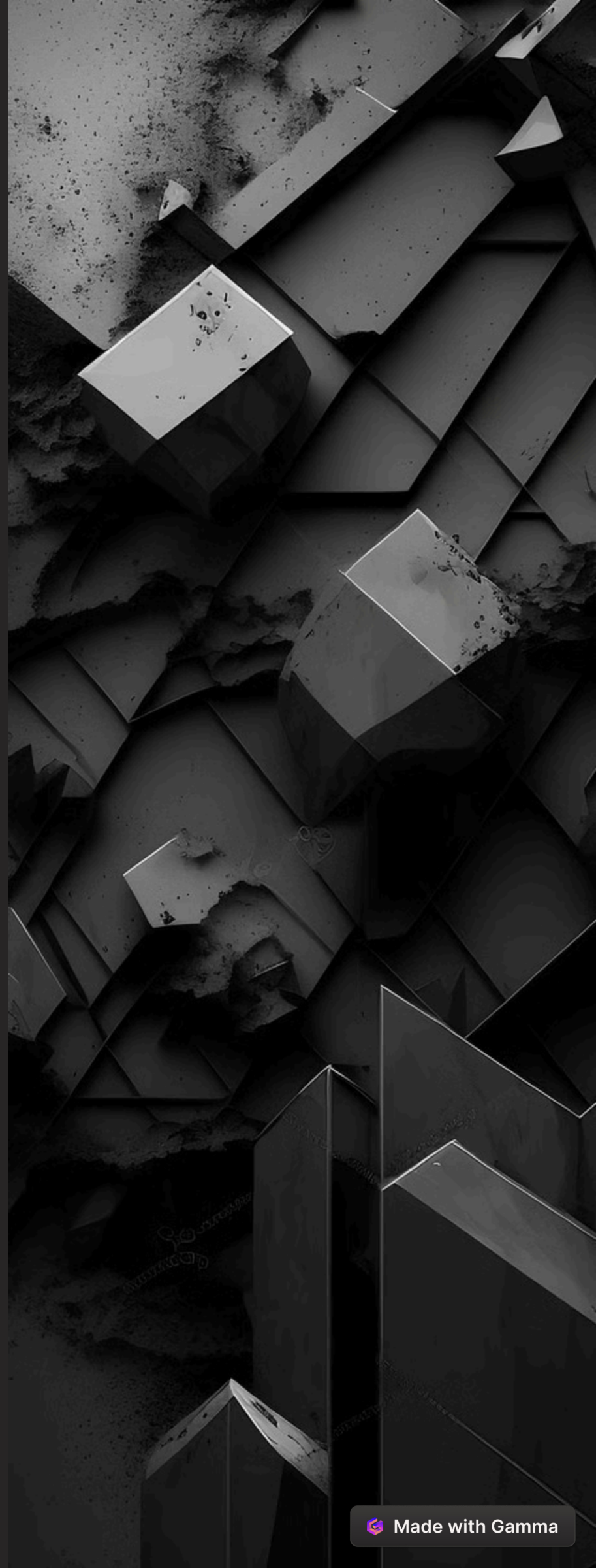
- Functions that take other functions as arguments or return functions.

```
function multiplyBy(num) {
  return function (x) {
    return x * num;
  };
}
const double = multiplyBy(2);
console.log(double(4)); // 8
```



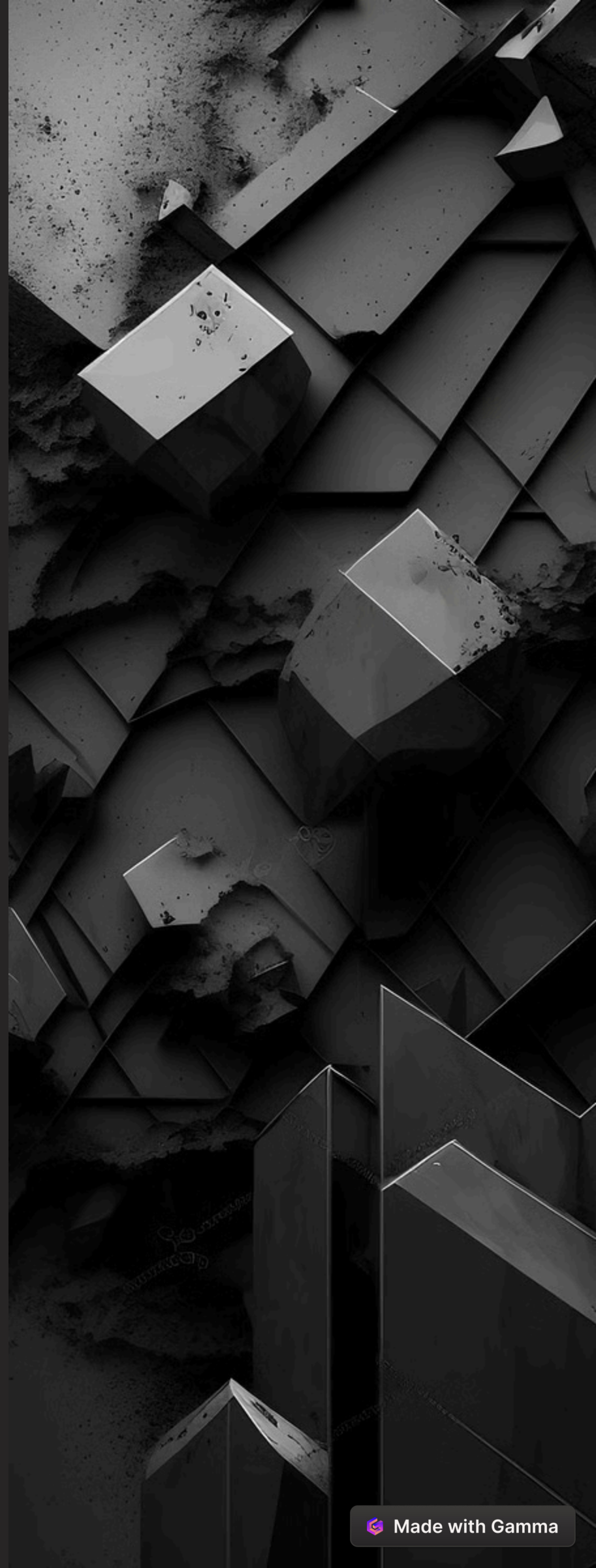
What are pure functions and why are they important?

- Functions that always return the same output for the same input and have no side effects.



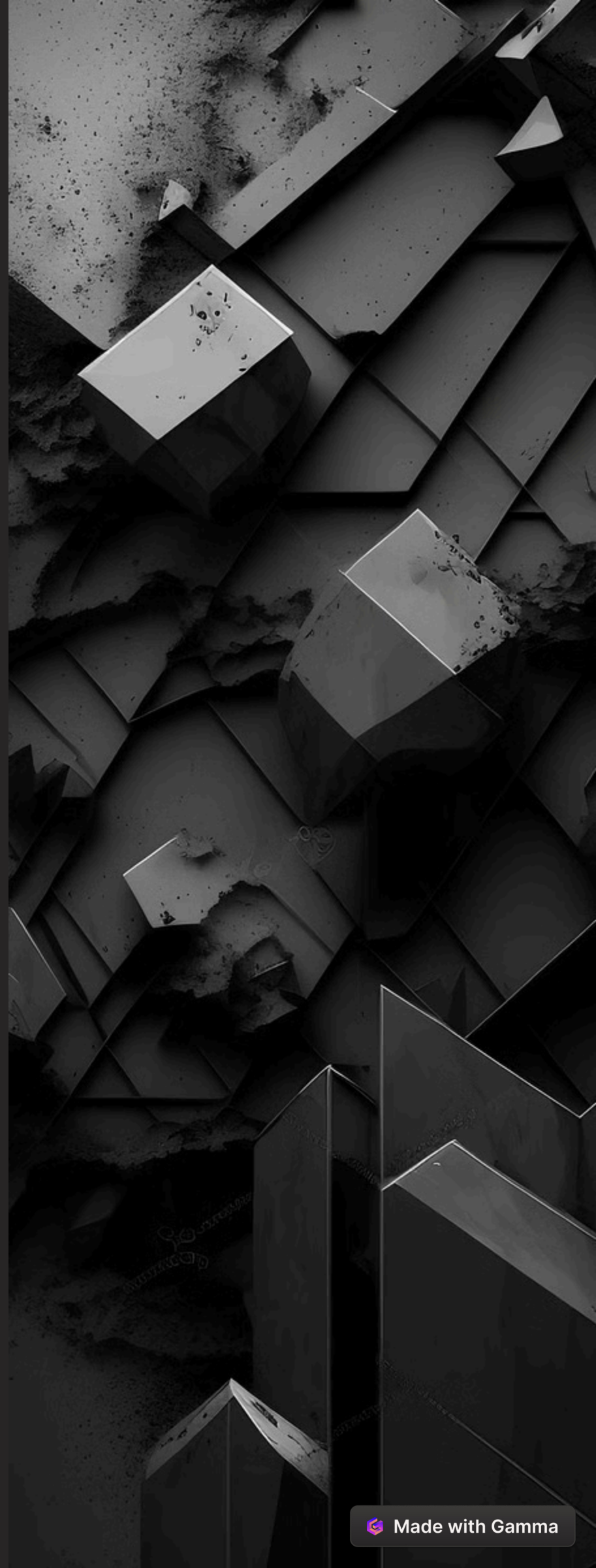
Explain call(), apply(), and bind() methods.

- call(): Calls a function with a given this value and arguments separately.
- apply(): Similar to call(), but takes arguments as an array.
- bind(): Returns a new function with this permanently set.



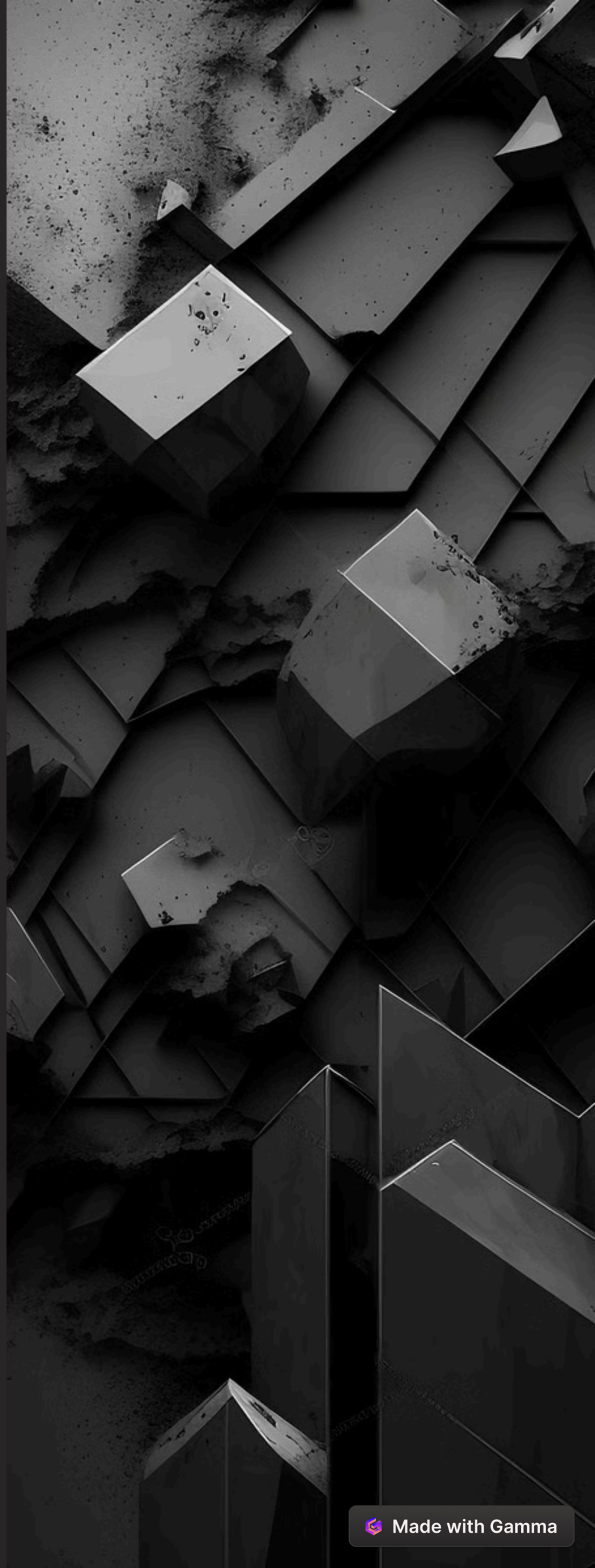
Difference between `==` and `===` in JavaScript.

- `==` checks value equality (type coercion allowed).
- `===` checks both value and type equality



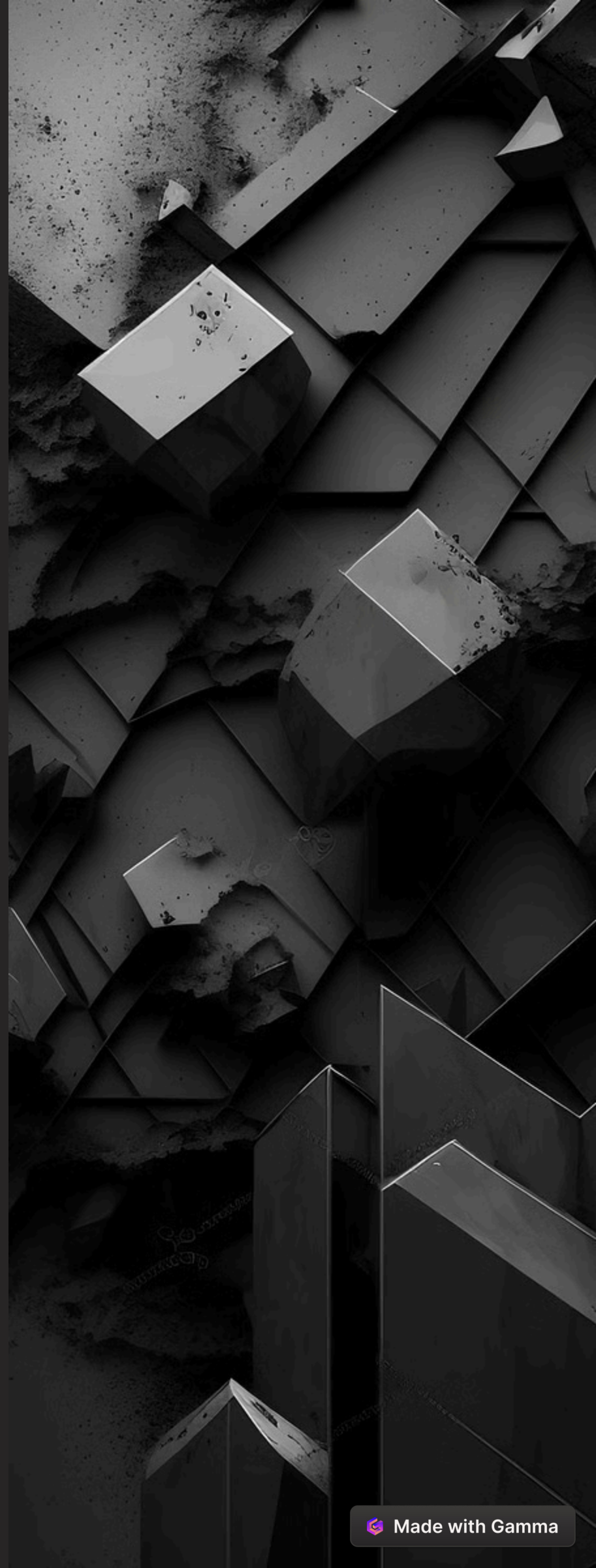
What is the difference between localStorage, sessionStorage, and cookies?

- localStorage: Data persists indefinitely.
- sessionStorage: Data persists until the session ends.
- cookies: Small data stored with an expiration date.



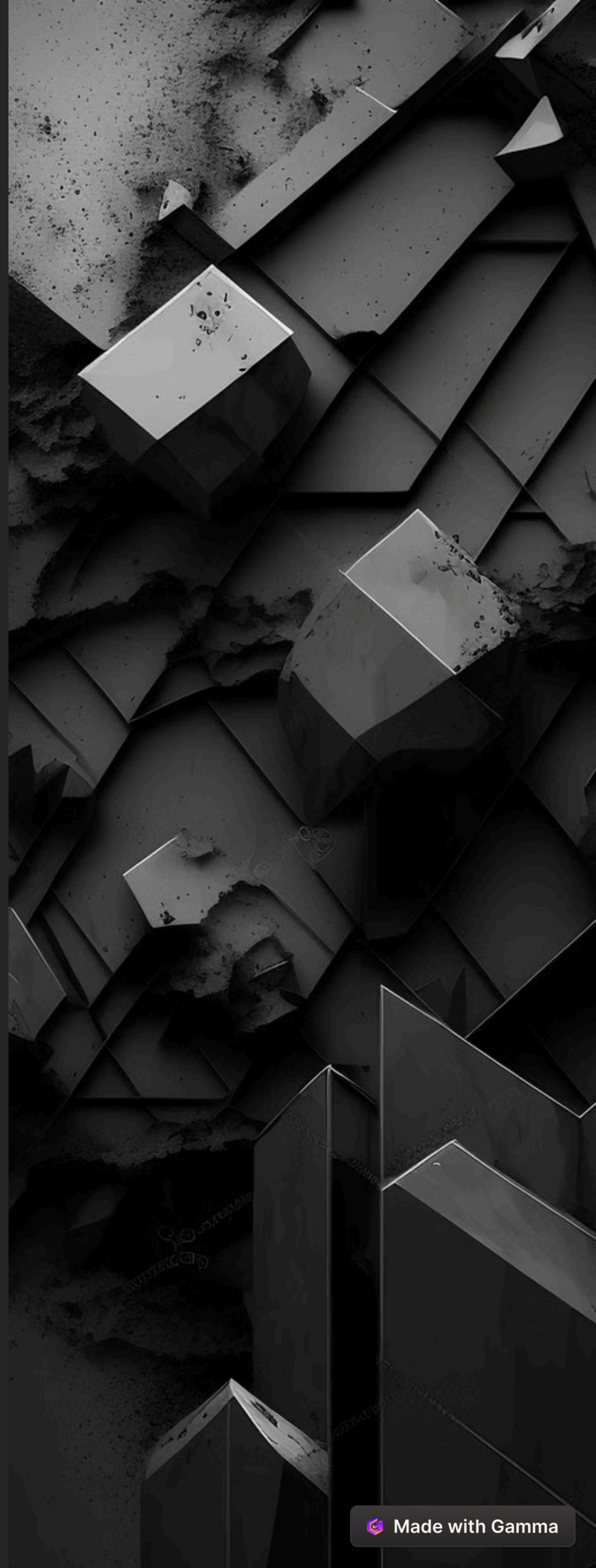
What are event delegation and event bubbling?

- Event delegation: Handling events at a parent level for better performance.
- Event bubbling: Event propagates from child to parent elements.



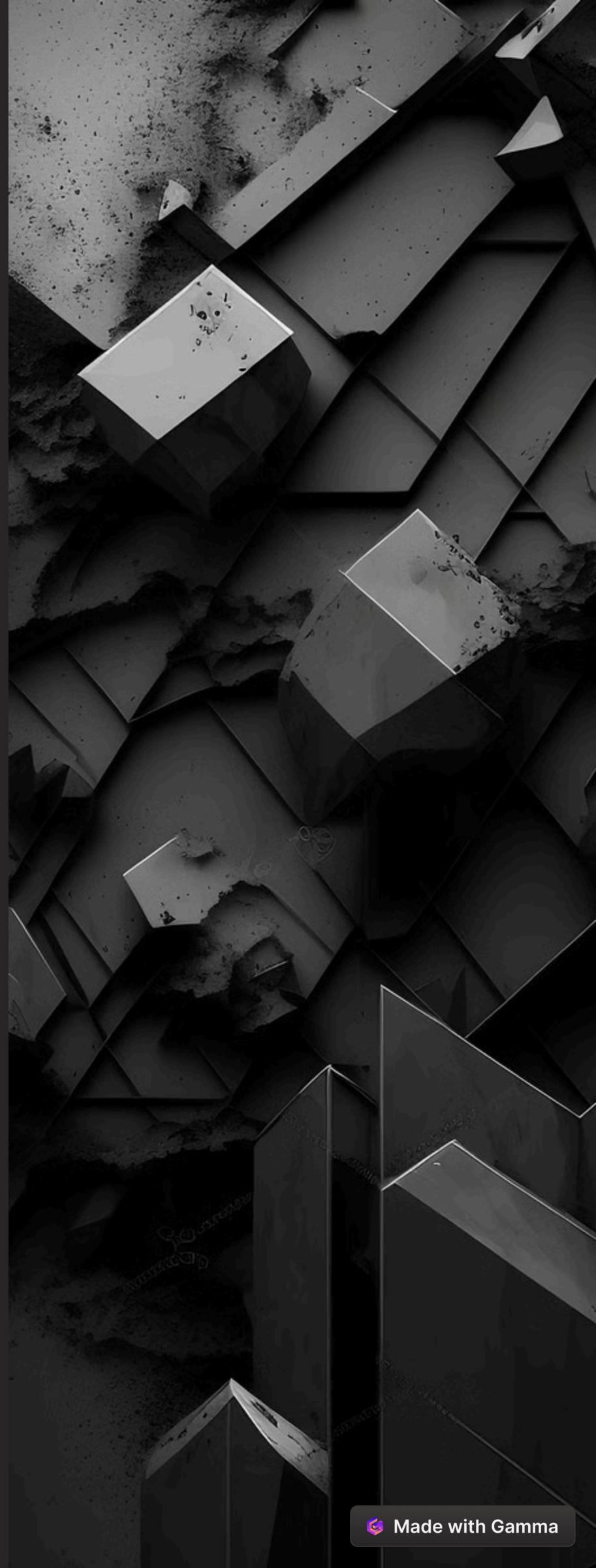
What is the purpose of the debounce and throttle functions?

- debounce(): Delays function execution until after a pause in events.
- throttle(): Ensures a function runs at most once in a specified time interval.



What is CORS, and why is it needed?

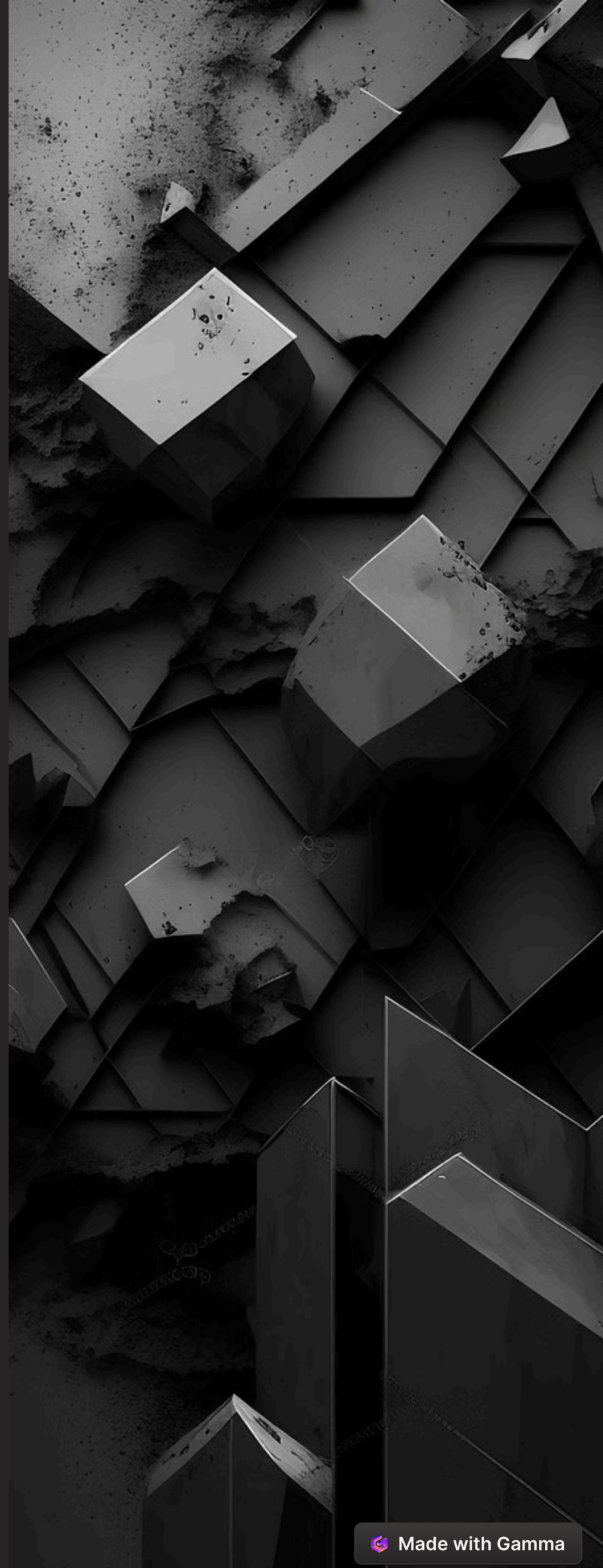
- Cross-Origin Resource Sharing (CORS) allows servers to control access to resources from different origins.



What are template literals in JavaScript?

- Strings enclosed with backticks `` ` `` that allow embedding expressions.

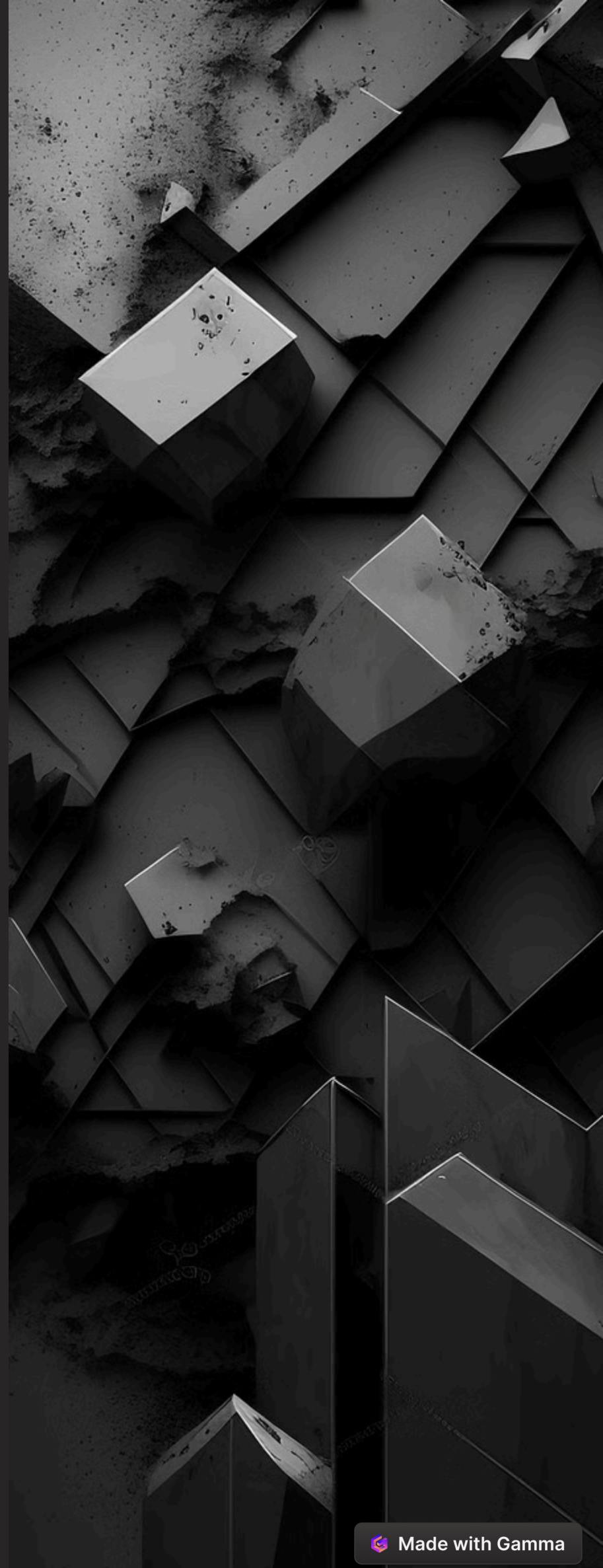
```
let name = "John";
console.log(`Hello, ${name}`);
```



What are destructuring and rest/spread operators?

- Destructuring: Extract values from arrays/objects.
- Rest `...`: Gathers remaining elements.
- Spread `...`: Expands elements.

```
let [a, b, ...rest] = [1, 2, 3, 4];
console.log(rest); // [3, 4]
```



What is the difference between `forEach()`, `map()`, `filter()`, and `reduce()`?

- `forEach()`: Iterates but does not return a new array.
- `map()`: Returns a new transformed array.
- `filter()`: Returns a new filtered array.
- `reduce()`: Reduces an array to a single value.

Conclusion

JavaScript, with its ever-evolving ecosystem and vibrant community, continues to shape the digital landscape. As we navigate through its intricacies, let's embrace the dynamic nature of JavaScript and its role in building the future of the web.