PEXELARE

Currying for JavaScript Devs

5 ways to use currying in your front-end flow.

PEXELARE

Case 1:
Dynamic Notification

Scenario:
You want to show different types of notifications (error, warning, info) with consistent formatting.

Where to use:
Toast messages, form validations, API error handlers.

```
const notify = type ⇒ message ⇒ `[${type.toUpperCase()}]: ${message}`;
const showError = notify("error");
console.log(showError("Something went wrong")); // [ERROR]: Something went
wrong
```

PEXELARE

Case 2:
Dynamic Text Formatter

Scenario:
User names from the database need to be consistently formatted (e.g., title case).

Where to use:
User profile, contact list, reviews/ testimonials.

```
const formatName = casing => name => {
  if (casing === "title") {
    return name
      .toLowerCase()
      .split(" ")
      .map(w => w.charAt(0).toUpperCase() + w.slice(1))
      .join(" ");
  }
  return name;
};
const toTitleCase = formatName("title");
console.log(toTitleCase("john DOE")); // John Doe
```

PEXELARE

Case 3: Password Validator

Scenario:
You want to validate passwords with customizable rules like minimum length and requiring numbers.

Where to use:
Sign-up pages, change password forms, account security settings.

```
const validatePassword = minLength ⇒ requireNumber ⇒
password ⇒ {
  const hasMinLength = password.length ⩾ minLength;
  const hasNumber = /\d/.test(password);
  return hasMinLength && (requireNumber ? hasNumber :
true);
};
const strongPassword = validatePassword(8)(true);
console.log(strongPassword("test1234")); // true
```

PEXELARE

Case 4:
CSS Class Generator

Scenario:
You want to generate class names dynamically based on component states or BEM structure.

Where to use:
Design systems, component libraries, dynamic UI rendering.

```
const createClass = base ⇒ modifier ⇒ `${base}--${modifier}`;
const buttonClass = createClass("btn");
console.log(buttonClass("primary")); // btn--primary
```

PEXELARE

Case 5:
Currency Formatter

Scenario:
Your website needs to display prices based on user's location and selected currency.

Where to use:
e-commerce websites,travel apps, hotel websites.

```
const formatCurrency = locale ⇒ currency ⇒ amount ⇒
  new Intl.NumberFormat(locale, { style: "currency", currency
}).format(amount);
const formatINR = formatCurrency("en-IN")("INR");
console.log(formatINR(1500)); // ₹1,500.00
```

PEXELARE

FOUND THIS HELPFUL?

FOLLOW FOR MORE.