Solution Review

A review of different solutions and their benefits/drawbacks. (3 min. read)

Here's our user, `bobo`.

```
const bobo = {
  firstName: 'Bobo',
  lastName: 'Flakes'
};
```

And we wanted to uppercase and reverse his first name.

The individual steps:

1. Get `firstName`
2. Uppercase it
3. Reverse it

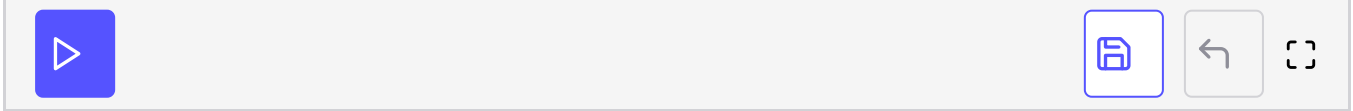
Method Chaining

Since all three steps deal with a string, we can use method chaining:

```
const upperAndReverseFirstName = (user) => {
  return user.firstName
    .toUpperCase()
    .split('')
    .reverse()
    .join('');
};

const result = upperAndReverseFirstName({
  firstName: 'Bobo'
});

console.log({ result });
```



This solution's a good example of composition and works fine.

This solution's a good example of composition and works fine.

It's a bit limited, however.
We'd likely find uppercasing and reversing strings useful somewhere else in our app! How can we better share that functionality?

Breaking It Up

Our steps can lead to three functions instead:

1. `getFirstName`
2. `uppercaseString`
3. `reverseString`

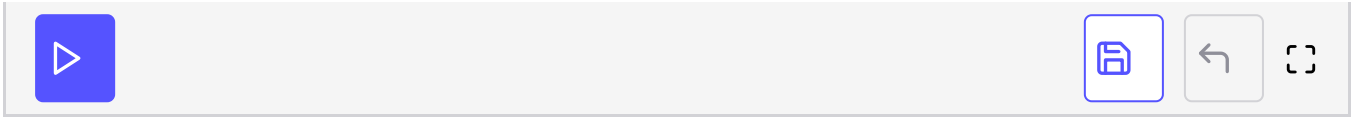
```
const getFirstName = (user) => user.firstName;
const uppercaseString = (string) => string.toUpperCase();
const reverseString = (string) => string
  .split('')
  .reverse()
  .join('');

const upperAndReverseFirstName = (user) => {
  const name = getFirstName(user);
  const uppercasedName = uppercaseString(name);

  return reverseString(uppercasedName);
};

const result = upperAndReverseFirstName({
  firstName: 'Bobo'
});

console.log({ result });
```



Now we have `upperAndReverseFirstName` and all the pieces it's composed of. Any other function can import the piece it needs, without hauling in everything else.

And if the requirements of a step change, we can just change the function responsible for that step without breaking the chain.

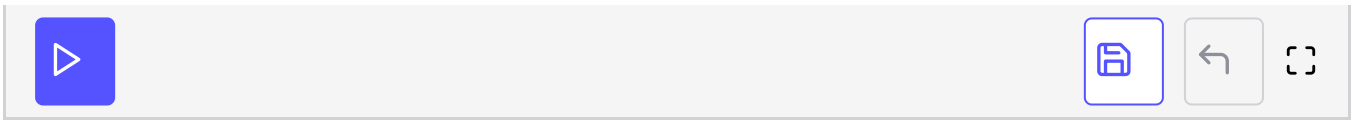
A List of Users

If you got that, then the second exercise was simple.

```
const upperAndReverseFirstNames = (users) => {
  return users.map(upperAndReverseFirstName);
};

const result = upperAndReverseFirstNames([{
  firstName: 'Bobo'
}, {
  firstName: 'Anon'
}, {
  firstName: 'Marko'
}]);

console.log({ result });
```



Compose `upperAndReverseFirstName` with `map` to make it applicable to Bobo and friends.