

JavaScript: a quick, simple and modern solution to get a unique array of objects

In JavaScript there is a quick, modern and simple solution to get a unique array of objects.

A modern solution involves the use of the `Set` object combined with the spread operator. This works well with arrays made solely of primitive data types. With objects, it requires a lot of further processing in order to achieve the desired result.

Since an array of strings is the ideal format for working with sets, we can turn every object literal into its JSON representation and then reconvert them into the original data type.

```
'use strict';

const arrOfObjects = [
  {
    id: 1,
    sku: 'ABC'
  },
  {
    id: 2,
    sku: 'CDE'
  },
  {
    id: 1,
    sku: 'ABC'
  },
]
```

```
    {
      id: 3,
      sku: 'LHX'
    },
    {
      id: 2,
      sku: 'CDE'
    },
    {
      id: 4,
      sku: 'ZKU'
    }
  ];

const serializeArr = arr => {
  return arr.map(obj => { return
JSON.stringify(obj); });
};

const arrayUnique = arr => {
  let objects = serializeArr(arr);
  let unique = [...new Set(objects)];
  return unique.map(str => { return
JSON.parse(str); } );
};
```

A simple test:

```
const test = () => {
  console.log('Initial array:');
  console.log(arrOf0bjects);
  console.log('Unique array:');
  console.log(arrayUnique(arrOf0bjects));
};
```

```
};
```

```
test();
```

Output:

Initial array:

```
[ { id: 1, sku: 'ABC' },  
  { id: 2, sku: 'CDE' },  
  { id: 1, sku: 'ABC' },  
  { id: 3, sku: 'LHX' },  
  { id: 2, sku: 'CDE' },  
  { id: 4, sku: 'ZKU' } ]
```

Unique array:

```
[ { id: 1, sku: 'ABC' },  
  { id: 2, sku: 'CDE' },  
  { id: 3, sku: 'LHX' },  
  { id: 4, sku: 'ZKU' } ]
```