# Obtaining CSS Design Values

This lesson shows how to programmatically access design values.

WE'LL COVER THE FOLLOWING ^



The `design-values` module in `css-theming` gives you programmatic access to some of the CSS defined variables. You can, for example, obtain and display all the semantic colors in your app:

```
import {
  getSemanticColorNames,
} from 'css-theming';

const semanticColorNames = getSemanticColorNames();
// Do whatever you want with `semanticColorNames`, it's an array of string
s.
```

> **Note:** All modules are exported in css-theming so you can import `from 'css-theming'` instead of `from 'design-values'`;

Here are all the definitions you can obtain from `design-values`:

```
swatchNames
fgSwatchNames
colorNames
semanticColorNames
themeCssNames
```

## Obtaining defined themes #

A common operation is displaying a list of your themes. For this we'll use the

`getThemes` function. This returns an array of `Theme` objects, defined as:

```
export interface Theme {
  category: string | null;
  name: string;
  cssName: string;
  brightness: Brightness;
}
```

So, `getThemes` might return something like:

```
[{
  "name": "default",
  "category": "",
  "cssName": "theme-default",
  "brightness": "light"
}, {
  "name": "default-dark",
  "category": "",
  "cssName": "theme-default-dark",
  "brightness": "dark"
}]
```

---

By getting design values in this way, we can easily create dynamic pages that show info about our theme, and more importantly, create a page where the user can select the theme. Next, we'll learn how to deal with multiple theme categories, something that a more complex app might need.