```
 1   import isArray from '../utils/is-array';
 2   import isObject from '../utils/is-object';
 3   import isObjectEmpty from '../utils/is-object-empty';
 4   import isUndefined from '../utils/is-undefined';
 5   import isNumber from '../utils/is-number';
 6   import isDate from '../utils/is-date';
 7   import map from '../utils/map';
 8   import { createInvalid } from './valid';
 9   import { Moment, isMoment } from '../moment/constructor';
10   import { getLocale } from '../locale/locales';
11   import { hooks } from '../utils/hooks';
12   import checkOverflow from './check-overflow';
13   import { isValid } from './valid';
14
15   import { configFromStringAndArray } from './from-string-and-array';
16   import { configFromStringAndFormat } from './from-string-and-format';
17   import { configFromString } from './from-string';
18   import { configFromArray } from './from-array';
19   import { configFromObject } from './from-object';
20
21   function createFromConfig(config) {
22       var res = new Moment(checkOverflow(prepareConfig(config)));
23       if (res._nextDay) {
24           // Adding is smart enough around DST
25           res.add(1, 'd');
26           res._nextDay = undefined;
27       }
28
29       return res;
30   }
31
32   export function prepareConfig(config) {
33       var input = config._i,
34           format = config._f;
35
36       config._locale = config._locale || getLocale(config._l);
37
38       if (input === null || (format === undefined && input === '')) {
39           return createInvalid({ nullInput: true });
40       }
41
42       if (typeof input === 'string') {
43           config._i = input = config._locale.preparse(input);
44       }
45
46       if (isMoment(input)) {
47           return new Moment(checkOverflow(input));
48       } else if (isDate(input)) {
49           config._d = input;
50       } else if (isArray(format)) {
51           configFromStringAndArray(config);
52       } else if (format) {
53           configFromStringAndFormat(config);
54       } else {
55           configFromInput(config);
56       }
57
58       if (!isValid(config)) {
59           config._d = null;
60       }
61
62       return config;
63   }
64
65   function configFromInput(config) {
66       var input = config._i;
67       if (isUndefined(input)) {
68           config._d = new Date(hooks.now());
69       } else if (isDate(input)) {
70           config._d = new Date(input.valueOf());
```

```
 71        } else if (typeof input === 'string') {
 72            configFromString(config);
 73        } else if (isArray(input)) {
 74            config._a = map(input.slice(0), function (obj) {
 75                return parseInt(obj, 10);
 76            });
 77            configFromArray(config);
 78        } else if (isObject(input)) {
 79            configFromObject(config);
 80        } else if (isNumber(input)) {
 81            // from milliseconds
 82            config._d = new Date(input);
 83        } else {
 84            hooks.createFromInputFallback(config);
 85        }
 86    }
 87
 88    export function createLocalOrUTC(input, format, locale, strict, isUTC) {
 89        var c = {};
 90
 91        if (format === true || format === false) {
 92            strict = format;
 93            format = undefined;
 94        }
 95
 96        if (locale === true || locale === false) {
 97            strict = locale;
 98            locale = undefined;
 99        }
100
101        if (
102            (isObject(input) && isObjectEmpty(input)) ||
103            (isArray(input) && input.length === 0)
104        ) {
105            input = undefined;
106        }
107        // object construction must be done this way.
108        // https://github.com/moment/moment/issues/1423
109        c._isAMomentObject = true;
110        c._useUTC = c._isUTC = isUTC;
111        c._l = locale;
112        c._i = input;
113        c._f = format;
114        c._strict = strict;
115
116        return createFromConfig(c);
117    }
```