To use `ReactiveVar`, add the `reactive-var` package to your project by running in your terminal:

```
meteor add reactive-var
```

{% apibox "ReactiveVar" %}

A ReactiveVar holds a single value that can be get and set, such that calling `set` will invalidate any Computations that called `get`, according to the usual contract for reactive data sources.

A ReactiveVar is similar to a Session variable, with a few differences:

- ReactiveVars don't have global names, like the "foo" in `Session.get('foo')`. Instead, they may be created and used locally, for example attached to a template instance, as in: `this.foo.get()`.
- ReactiveVars are not automatically migrated across hot code pushes, whereas Session state is.
- ReactiveVars can hold any value, while Session variables are limited to JSON or EJSON.

An important property of ReactiveVars — which is sometimes a reason for using one — is that setting the value to the same value as before has no effect; it does not trigger any invalidations. So if one autorun sets a ReactiveVar, and another autorun gets the ReactiveVar, a re-run of the first autorun won't necessarily trigger the second. By default, only primitive values are compared this way, while calling `set` on an argument that is an *object* (not a primitive) always counts as a change. You can configure this behavior using the `equalsFunc` argument.

{% apibox "ReactiveVar#get" %}

{% apibox "ReactiveVar#set" %}