SVELTE • REFERENCE

svelte/store

ON THIS PAGE

```
import {
  derived,
  fromStore,
  get,
  readable,
  readonly,
  toStore,
  writable
} from 'svelte/store';
```

derived

Derived value store by synchronizing one or more readable stores and applying an aggregation function over its input values.

```
function derived<S extends Stores, T>(
  stores: S,
  fn: (
    values: StoresValues<S>,
    set: (value: T) => void,
    update: (fn: Updater<T>) => void
  ) => Unsubscriber | void,
  initial_value?: T | undefined
): Readable<T>;
```

```
function derived<S extends Stores, T>(
  stores: S,
  fn: (values: StoresValues<S>) => T
```

fromStore

```
function fromStore<V>(store: Writable<V>): {
  current: V;
};
```

```
function fromStore<V>(store: Readable<V>): {
  readonly current: V;
};
```

get

Get the current value from a store by subscribing and immediately unsubscribing.

```
function get<T>(store: Readable<T>): T;
```

readable

Creates a `Readable` store that allows reading by subscription.

```
function readable<T>(
  value?: T | undefined,
  start?: StartStopNotifier<T> | undefined
): Readable<T>;
```

readonly

toStore

```
function toStore<V>(
  get: () => V,
  set: (v: V) => void
): Writable<V>;
```

```
function toStore<V>(get: () => V): Readable<V>;
```

writable

Create a `Writable` store that allows both updating and reading by subscription.

```
function writable<T>(
  value?: T | undefined,
  start?: StartStopNotifier<T> | undefined
): Writable<T>;
```

Readable

Readable interface for subscribing.

```
interface Readable<T> {…}
```

```
subscribe(this: void, run: Subscriber<T>, invalidate?: () => void): Unsubscriber;
```

`run` subscription callback

`invalidate` cleanup callback

StartStopNotifier

Start and stop notification callbacks. This function is called when the first subscriber subscribes.

```
type StartStopNotifier<T> = (
  set: (value: T) => void,
  update: (fn: Updater<T>) => void
) => void | (() => void);
```

Subscriber

Callback to inform of a value updates.

```
type Subscriber<T> = (value: T) => void;
```

Unsubscriber

Unsubscribes from value updates.

```
type Unsubscriber = () => void;
```

Updater

Callback to update a value.

```
type Updater<T> = (value: T) => T;
```

Writable interface for both updating and subscribing.

```
interface Writable<T> extends Readable<T> {…}
```

```
set(this: void, value: T): void;
```

`value` to set

Set value and inform subscribers.

```
update(this: void, updater: Updater<T>): void;
```

`updater` callback

Update value using callback and inform subscribers.

Edit this page on GitHub

PREVIOUS
svelte/server

NEXT
svelte/transition