SVELTE • REFERENCE

Runtime warnings

ON THIS PAGE

Client warnings

binding_property_non_reactive

```
`%binding%` is binding to a non-reactive property
```

```
`%binding%` (%location%) is binding to a non-reactive property
```

console_log_state

```
Your `console.%method%` contained `$state` proxies. Consider using `$inspect(...)` or `$s
```

When logging a proxy, browser devtools will log the proxy itself rather than the value it represents. In the case of Svelte, the 'target' of a `$state` proxy might not resemble its current value, which can be confusing.

The easiest way to log a value as it changes over time is to use the `$inspect` rune. Alternatively, to log things on a one-off basis (for example, inside an event handler) you can use `$state.snapshot` to take a snapshot of the current value.

event_handler_invalid

hydration_attribute_changed

```
The `%attribute%` attribute on `%html%` changed its value between server and client rende
```

hydration_html_changed

```
The value of an `{@html ...}` block changed between server and client renders. The client
```

```
The value of an `{@html ...}` block %location% changed between server and client renders.
```

hydration_mismatch

```
Hydration failed because the initial UI does not match what was rendered on the server
```

```
Hydration failed because the initial UI does not match what was rendered on the server. Th
```

invalid_raw_snippet_render

```
The `render` function passed to `createRawSnippet` should return HTML for a single elemen
```

legacy_recursive_reactive_block

```
Detected a migrated `$:` reactive block in `%filename%` that both accesses and updates the
```

ownership_invalid_binding

```
%parent% passed a value to %child% with `bind:`, but the value is owned by %owner%. Consi
```

ownership_invalid_mutation

```
Mutating a value outside the component that created it is strongly discouraged. Consider
```

```
%component% mutated a value owned by %owner%. This is strongly discouraged. Consider pass
```

state_proxy_equality_mismatch

```
Reactive `$state(...)` proxies and the values they proxy have different identities. Becau
```

`$state(...)` creates a proxy of the value it is passed. The proxy and the value have different identities, meaning equality checks will always return `false` :

```
<script>
  let value = { foo: 'bar' };
  let proxy = $state(value);

  value === proxy; // always false
</script>
```

To resolve this, ensure you're comparing values where both values were created with `$state(...)` , or neither were. Note that `$state.raw(...)` will *not* create a state proxy.

```
`<svelte:element this="%tag%">` is a void element — it cannot have content
```

state_snapshot_uncloneable

```
Value cannot be cloned with `$state.snapshot` — the original value was returned
```

```
The following properties cannot be cloned with `$state.snapshot` — the return value conta

%properties%
```

Edit this page on GitHub