How to query elements within shadow DOM from outside in Dart?

Asked 9 years, 11 months ago Modified 1 year, 3 months ago Viewed 96k times

50

How can I select nodes within shadow DOM? Consider the following example:

structure of "unshadowed" DOM

```
<app-element>
  #shadow-root
    <h2></h2>
    <content>
      #outside shadow
      <h2></h2>
    </content>
    <ui-button>
      #shadow-root
        <h2></h2>
  </ui-button>
</app-element>
```

index.html

```
<body>
<app-element>
  <!-- OK: querySelect('app-element').querySelect('h2') -->
  <!-- OK: querySelect('app-element h2') -->
  <!-- There is no problem to select it -->
  <h2>app-element > content > h2</h2>
</app-element>
</body>
```

templates.html

```
<polymer-element name="ui-button" noscript>
  <template>
    <!-- FAIL: querySelect('app-element::shadow ui-button::shadow h2') -->
    <h2>app-element > ui-button > h2</h2>
  </template>
</polymer-element>

<polymer-element name="app-element" noscript>
  <template>
    <!-- FAIL: querySelect('app-element::shadow').querySelect('h2') -->
    <!-- FAIL: querySelect('app-element::shadow h2') -->
    <!-- FAIL: querySelect('app-element').shadowRoot.querySelect('h2') -->
    <h2>app-element > h2</h2>
    <content></content>
    <ui-button></ui-button>
  </template>
</polymer-element>
```

In comments like "OK: querySelect()" I show selectors I've tried to run from outside any shadowed DOM.

I've already read the following article: http://www.html5rocks.com/en/tutorials/webcomponents/shadowdom-201/?redirect_from_locale=ru and based on the fact that it was said in the article, query like: `document.querySelector('app-element::shadow h2');` in JS should work as expected. However in Dart it doesn't work.

What do I wrong?

css-selectors dart polymer dart-polymer shadow-dom

Share Improve this question Follow

edited Apr 15, 2015 at 7:53

asked Apr 14, 2015 at 14:04

Timur Fayzrakhmanov
19.7k 20 69 102

After a looong time i wrote a **general solution** for this. See my answer: **stackoverflow.com/a/75241202/378024** – gblazex Jan 26, 2023 at 0:16

7 Answers

Sorted by: Highest score (default) ⬍

Pseudo selector `::shadow` and combinator `/deep/` doesn't work on firefox.

60

Use `.shadowRoot`

```
var shadowroot = app-element.shadowRoot;
shadowroot.querySelector('h2');
```

Share Improve this answer Follow

answered Jun 14, 2016 at 12:05

Aniruddha
806 1 6 12

4 And were removed in Chrome 63. – Steven Vachon May 22, 2019 at 17:29

I want to add that this works for accessing child elements within your StencilJS components too! So in my case, it looks like this: `const myCarousel = this.host.shadowRoot.querySelector( #myCarousel );` where host is `@Element() host: HTMLElement;` . Thanks! – Omar Mar 29, 2021 at 20:59

this works, but if it's deeply nested it can be tedious to run many selectors inside many shadow roots. Luckily there is a way to just write 2 general loops nested. See: stackoverflow.com/a/75241202/378024 – gblazex Jan 26, 2023 at 0:17

16

For people wanting an easy to use solution

```
function $$$(selector, rootNode=document.body) {
    const arr = []

    const traverser = node => {
        // 1. decline all nodes that are not elements
        if(node.nodeType !== Node.ELEMENT_NODE) {
            return
        }

        // 2. add the node to the array, if it matches the selector
        if(node.matches(selector)) {
            arr.push(node)
        }

        // 3. loop through the children
        const children = node.children
        if (children.length) {
            for(const child of children) {
                traverser(child)
            }
        }

        // 4. check for shadow DOM, and loop through it's children
        const shadowRoot = node.shadowRoot
        if (shadowRoot) {
            const shadowChildren = shadowRoot.children
            for(const shadowChild of shadowChildren) {
                traverser(shadowChild)
            }
        }
    }

    traverser(rootNode)

    return arr
}
```

Use it like this:

```
var nodes = $$$('#some .selector')

// use from a custom rootNode
var buttonsWithinFirstNode = $$$('button', nodes[0])
```

It will traverse all the elements within the rootNode, so it won't be fast but it is easy to use.

Share Improve this answer Follow

answered Apr 7, 2022 at 19:14

nathnolt
545 6 10

1 It's better to use `rootNode=document.documentElement` than `rootNode=document.body` . – Niklas Jul 18, 2022 at 17:19

1 If you set rootNode=document.documentElement, You can find the elements inside of <head> as well. – nathnolt Jul 20, 2022 at 17:46

couldn't benchmark, but works fast, as it essentially reimplements `querySelectorAll` method with additional shadowRoot children parsing. NOTE: this is amazing for testing and scraping, but for dev such a style largely defeats the purpose of shadow root. Use in moderation (if at all). – Kukuster Aug 22, 2023 at 8:27

@NiklasE. What is the use case for having elements with a shadowRoot in the head? I can't think of one, and would think it just makes the search slower. – run_the_race Oct 23, 2023 at 11:18

@run_the_race, chrome extensions. – Vlas Bashynskyi Feb 7 at 12:25

14

Outdated

ShadowDom was changed significantly since I posted this answer, this is why this answer doesn't apply anymore.

Original Answer

If you use a custom main, ensure that Polymer is properly initialized before you try to interact with your Polymer elements (see how to implement a main function in polymer apps for more details).

I usually suggest to avoid a custom main and create an `app-element` (or whatever name you prefer) and put your initialization code into `attached` (ensure to call `super.attached();`) or in `ready()` (doesn't need the super call).

It seems in this case it's not in the shadow DOM but a child.

This should work:

```
querySelector('h2');
```

It's only in the shadow DOM when it is within your elements `<template>...</template>` not when you wrap it in the tag of your custom element.

```
<polymer-element name="some-element">
  <template>
    <!-- this becomes the shadow DOM -->
    <content>
     <!--
       what gets captureD by the content element becomes a child or some-element
       -->
     </content>
  </template>
</polymer-element>



<body>
  <some-element>
    <!-- these elements here are captured by the
         content tag and become children of some-element -->
    <div>some text</div>
  </some-element>
</body>
```

If you want to search

inside the shadow DOM of the current element

```
shadowRoot.querySelect('h2');
```

inside the shadow DOM of an element inside the shadow DOM

```
shadowRoot.querySelector('* /deep/ h2');
shadowRoot.querySelector('ui-button::shadow h2');
```

from outside the current element

```
import 'dart:html' as dom;
...
dom.querySelector('* /deep/ h2');
// or (only in the shadow DOM of <app-element>)
dom.querySelector('app-element::shadow h2');
dom.querySelector('app-element::shadow ui-button::shadow h2');
// or (arbitrary depth)
dom.querySelector('app-element /deep/ h2');
```

Share Improve this answer Follow

edited Nov 15, 2023 at 7:02

answered Apr 14, 2015 at 14:15

Günter Zöchbauer
658k 233 2.1k 1.6k

1 Where is your `querySelector...` code? In main, in the `AppElement` class, where in the class? – Günter Zöchbauer Apr 14, 2015 at 18:30

Using CSS everything works as expected: pseudo selector like `::shadow` and combinator `/deep/` affect the elements. However it doesn't make sense when I use Dart's `querySelect(All)` functions. Only h2 inside `<content>` tag is reachable. What's wrong? – Timur Fayzrakhmanov Apr 14, 2015 at 18:35

Code in the `main() {}` – Timur Fayzrakhmanov Apr 14, 2015 at 18:35

12 `::shadow` and `/deep/` were removed in Chrome 63. – Steven Vachon May 22, 2019 at 17:29

3

Slightly tweaked @nathnolt's nice solution so one has two functions:

- `queryDeepSelectorAll`
- `queryDeepSelector` for querying a single element, it's faster to return as soon as a match is found.

```
/* Queries the document and shadowRoot
 * Based on https://stackoverflow.com/a/71787772/5506400
 * @param: all - bool
 *  `true`  - works like querySelectorAll,
 *  `false` - works like querySelector,
 */
export function queryDeepSelectorAll(selector, rootNode=document.body, all=true)
{
    const arr = []
    const traverser = node => {
        if (!all && arr.length) return;

        // 1. decline all nodes that are not elements
        if(node.nodeType !== Node.ELEMENT_NODE) return

        // 2. add the node to the array, if it matches the selector
        if(node.matches(selector)) {
            arr.push(node)
            if (!all) return;
        }
        // 3. loop through the children
        const children = node.children
        if (children.length) {
            for(const child of children) {
                traverser(child)
            }
        }
        // 4. check for shadow DOM, and loop through it's children
        const shadowRoot = node.shadowRoot
        if (shadowRoot) {
            const shadowChildren = shadowRoot.children
            for(const shadowChild of shadowChildren) {
                traverser(shadowChild)
            }
        }
    }
    traverser(rootNode)
    return all ? arr : ( arr.length ? arr[0] : null );
}
export function queryDeepSelector(selector, rootNode=document.body) {
    return queryDeepSelectorAll(selector, rootNode, false);
}
```

I do **not** recommend adding these functions to the HTMLElement prototype as a side effect when a module is loaded, side effects will cause all sorts of issues when refactoring as they hide dependencies.

```
// NOT recommended:
// Add queryDeepSelectorAll method to the HTMLElement prototype
if (!HTMLElement.prototype.queryDeepSelectorAll) {
    HTMLElement.prototype.queryDeepSelectorAll = function(selector) {
        const arr = [];
        const traverser = node => {
            ...
        };
        traverser(this);
        return arr;
    };
}
```

Share Improve this answer Follow

answered Oct 23, 2023 at 12:45

run_the_race
2,418 3 54 84

1

This is an old question but I was amazed that there is no general solution (not just in Dart). Let's solve it for **all cases**!

```
// query elements even deeply within shadow doms. e.g.:
// ts-app::shadow paper-textarea::shadow paper-input-container
function querySelectorDeep(selector, root = document) {
  let currentRoot = root;
  let partials = selector.split('::shadow');
  let elems = currentRoot.querySelectorAll(partials[0]);
  for (let i = 1; i < partials.length; i++) {
    let partial = partials[i];
    let elemsInside = [];
    for (let j = 0; j < elems.length; j++) {
      let shadow = elems[j].shadowRoot;
      if (shadow) {
        const matchesInShadow = shadow.querySelectorAll(partial);
        elemsInside = elemsInside.concat([... matchesInShadow]);
      }
    }
    elems = elemsInside;
  }
  return elems;
}
```

Example to try it out:

1. Go to Google's text-to-speech demo
2. Open Console and enter:

```
let sel = `ts-app::shadow
           paper-textarea::shadow
           paper-input-container
           iron-autogrow-textarea::shadow
           textarea`;
textarea = querySelectorDeep(sel)?.[0];
textarea.value = 'If you see this, selector worked.';
```

Note: This could be solved recursively as well, but I just went with iterative.

Share Improve this answer Follow

edited Dec 7, 2023 at 17:23

answered Jan 26, 2023 at 0:14

gblazex
50.1k 12 99 92

0

Vanilla only helper function using reduce method

```
function queryShadow([firstShadowSelector, ...restOfTheShadowSelectors], itemSelector) {
    const reduceFunction = (currShadow, nextShadowSelector) => currShadow.shadowRoot.querySelector(nextShadowSelector);
    const firstShadow = document.querySelector(firstShadowSelector);
    const lastShadow = restOfTheShadowSelectors.reduce(reduceFunction,firstShadow);
    return lastShadow && lastShadow.querySelector(itemSelector);
}
```

and use it like this

```
const shadowSelectorsArr = ['vt-virustotal-app','file-view', '#report', 'vt-ui-file-card', 'vt-ui-generic-card'];
const foundDomElem = queryShadow(shadowSelectorsArr, '.file-id');
console.log(foundDomElem && foundDomElem.innerText);
```

Share Improve this answer Follow

edited Mar 27, 2020 at 21:44

marc_s
756k 184 1.4k 1.5k

answered Mar 23, 2020 at 14:35

yoni12ab
21 2

0

The previous solutions didn't work for me, but this did:

```
function $$(selector, element = document.body) {
  if (element.shadowRoot) {
    element = element.shadowRoot;
    const found = element.querySelector(selector);
    if (found) return found;
  }
  for (let i = 0; i < element.children.length; i++) {
    const child = element.children[i];
    const found = $$(selector, child);
    if (found) {
      return found;
    }
  }
  return null;
}
```

use like `$$('.my-text-box')`

Share Improve this answer Follow

edited Apr 25, 2023 at 5:04
Eric Aya
70.1k 36 189 259

answered Apr 25, 2023 at 0:48
Leah Zorychta
13.5k 7 47 83

Start asking to get answers

Find the answer to your question by asking.

Ask question

Explore related questions

css-selectors dart polymer dart-polymer shadow-dom

See similar questions with these tags.