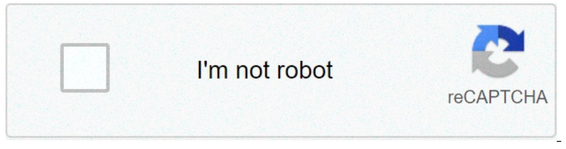

Continue

Css for first child

选择属于其父元素的首个子元素的每个 元素，并为其设置样式： p:first-child { background-color:yellow; } 亲自试一试 选择器 Chrome IE Firefox Safari Opera :first-child 4.0 7.0 3.0 3.1 9.6 注释：对于 IE8 及更早版本的浏览器中的 :first-child，必须声明 。 :first-child 选择器用于选取属于其父元素的首个子元素的指定选择器。 选择每个 中的每个 元素并设置其样式，其中的 元素是其父元素的第一个子元素： p:first-child i { background:yellow; } 亲自试一试 例子 2 选择列表中的第一个 元素并设置其样式： li:first-child { background:yellow; } 亲自试一试 例子 3 设置每个 的首个子元素，并设置其样式： ul>:first-child { background:yellow; } 亲自试一试 :first-child CSS pseudo-class 表示在一组兄弟元素中的第一个元素。 p:first-child { color: lime; } Note: 最初定义时，所选元素必须有一个parent。而从选择器 Level 4 开始，parent不再是必须的。 This text is selected! This text isn't selected. This text isn't selected: it's not a `p`. This text isn't selected. p:first-child { color: lime; background-color: black; padding: 5px; } Item 1 Item 2 Item 3 Item 3.1 Item 3.2 Item 3.3 ul li { color: blue; } ul li:first-child { color: red; font-weight: bold; } BCD tables only load in the browser相关链接 :first-of-type, :last-child, :nth-child The :first-child CSS pseudo-class represents the first element among a group of sibling elements. p:first-child { color: lime; } Note: As originally defined, the selected element had to have a parent. Beginning with Selectors Level 4, this is no longer required. This text is selected! This text isn't selected. This text isn't selected: it's not a `p`. This text isn't selected. p:first-child { color: lime; background-color: black; padding: 5px; } Item 1 Item 2 Item 3 Item 3.1 Item 3.2 Item 3.3 ul li { color: blue; } ul li:first-child { color: red; font-weight: bold; } SpecificationSelectors Level 4 (Selectors 4)# first-child-pseudoBCD tables only load in the browserSee also :-moz-first-node :first-of-type :last-child :nth-child() WEB前端开发 » CSS3参考手册 » 选择符列表 » 伪类选择符 » 匹配父元素的第一个子元素E。 要使该属性生效，E元素必须是某个元素的子元素，E的父元素最高是body，即E可以是body的子元素 这里可能存在误解： 示例代码： 在上述代码中，如果我们要设置第一个li的样式，那么代码应该写成li:first-child{sRules}，而不是ul:first-child{sRules}。 来看这样一段代码： 假设将代码简单地修改一下： 示例代码： p:first-child{color:#f00;} 只是在p前面加了一个h2标签，你会发现选择器失效了，没有命中p，why? first-child选择符，E必须是它的兄弟元素中的第一个元素，换言之，E必须是父元素的第一个子元素。与之类似的伪类还有last-child，只不过情况正好相反，需要它是最后一个子元素。 浅绿 = 支持 红色 = 不支持 粉色 = 部分支持 IE Firefox Safari Chrome Opera 6.0 4.0 4.0 4.0 15.0 7.0 以上就是这篇first-child - CSS :first-child选择器-CSS伪类选择符的全部内容，更多文章请进入前端开发博客 除了html5的新特性，CSS3的新特性也是面试中经常被问到的。本文分享了一些CSS3选择器、Transition,Transform和Animation等 - 2017-07-09vw：1vw 等于视口宽度的1%，vh：1vh 等于视口高度的1%。本文介绍纯CSS视口单位vw和vh来自行自适应，虽然现在的兼容性还没法完全能够接受，但不妨碍你认识这个vw和vh的强大。 - 2017-07-26CSS3 的 calc() 函数允许我们在属性值中执行数学计算操作。例如，我们可以使用 calc() 指定一个元素宽的固定像素值为多个数值的和。本文分析了calc()的计算使用方法及兼容性 - 2017-05-10一个使用伪元素来实现边框逐渐发光的过渡效果，主要用到scale和opacity这两个属性。 - 2017-03-31这个 CSS3 类似于幻灯片旋转的效果，是一个比较有意思并且比较受欢迎的特效。之前没有去研究过，无意在博客上看到 Wenzi 写了这个东西，来看看他的代码是怎么实现的。 - 2017-03-21作者从CSS3动画的基础入手，分别介绍了移动、缩放、旋转、扭曲到矩阵的变形。在最后给我们讲了关于CSS3矩阵的深度问题研究，值得看看。 - 2017-02-12JavaScript判断浏览器是否支持CSS3属性在使用CSS3的一些属性时，为了兼顾低端浏览器对CSS3的不友好性，往往需要知道某些浏览器是否支持要使用的CSS3属性，以此来做向下适配。比如常见的CSS3动画就很有必要检测浏览器是否支持。下面分享几种方法： - 2016-09-24任何CSS属性值为percent时，都需要根据某个参考值进行计算，搞明白这个参考值是什么，理解就容易多了。标准规定：background-position:perenct的参考值为: (容器宽度 - 背景图片宽度). - 2016-06-24 CSS3参考手册，全网最新最全的CSS3参考手册，为你呈现最好的CSS3文档CSS3参考手册 :first-child 选择器用于选取属于其父元素的首个子元素的指定选择器。——w3school 嗯，乍一看好像说的不是很明白，因此这个选择器很容易让人误解，通常会有两种误解：误解一：认为E:first-child选中E元素的第一个子元素。误解二：认为E:first-child选中E元素的父元素的第一个E元素。 你是不是也曾这样理解这个选择器或者现在仍然这样理解？以上两种理解都是错误的，为了证明上面两种理解是错的，看看下面的实例 div:first-child{color: red;} 一个链接 一个链接 一个链接 一个链接 效果是这样的： 很明显，照第一种理解，应该只有第一个a元素字体颜色变红，然而事实上全部变红了。 div a:first-child{color: red;} 一个段落 一个链接 一个链接 一个链接 一个链接 效果是这样的： 照第二种理解，div里的第一个a元素字体应该是红色的，事实证明这种理解也是错的。 OK，正确的理解应该是：只要E元素是它的父级的第一个子元素，就选中。它需要同时满足两个条件，一个是“第一个子元素”，另一个是“这个子元素刚好是E”。 span:first-child{color: red;} p:first-child{color: blue;} /*p元素的父元素的第一个子元素是div而不是p元素，因此该样式不起作用*/ i:first-child{color: orange;} .demo的第一个子元素是div 第一个span第一个段落P第二个span 一个链接第一个i元素 一个链接第二个i元素 一个链接 效果： 类似容易误解的结构选择器还有:nth-child()、:nth-last-child、:only-child，在平时的开发中需要注意一下。 E:first-child是伪类选择器， 匹配父元素的第一个子元素E 从说明可以看出E是你要选择的第一个子元素，而不是父元素。一开始，误以为E:first-child是E的第一个子元素。 匹配父元素的第n个子元素E E也是子元素，而且也只能匹配父元素下第n个子元素。n是从1开始计数 l1 l2 l3 若要选择 l1 ul>li:first-child 若要选择 l2 ul>li:nth-child(2) h1 p1 p2 p3 这时选择第一个p元素，应用p:first-child则会出现错误，因为p的父元素是div，而对于div来说，它的第一个子元素不是p，而是h1，所以如果选择器p:first-child，则会出错。 同理，E:last-child``E:only-child与上面的一样，E元素必须是其父元素的最后一个子元素或唯一一个子元素才可以 What you posted literally means "Find any divs that are inside of section divs and are the first child of their parent." The sub contains one tag that matches that description. It is unclear to me whether you want both children of the main div or not. If so, use this: div.section > div If you only want the header, use this: div.section > div:first-child Using the > changes the description to: "Find any divs that are the direct descendents of section divs" which is what you want. Please note that all major browsers support this method, except IE6. If IE6 support is mission-critical, you will have to add classes to the child divs and use that, instead. Otherwise, it's not worth caring about. Introduction: 介绍： Well, selectors are a very common term to deal with while we are developing a website or web page. You might know quite a few of them and might as well be implementing them. You might also have noticed that all the selectors are used for selecting some element or other, well that's their entire purpose and thus the name selectors. Although all the selectors share common characteristics but not all of them are the same. Their behaviors differ to a very great extent and one can only know after practical implementation. So, since we are discussing selectors, why don't we talk about one very specific selector? Although you might have come across this selector at some point of your time. The selector we will be focused on is not:first-child selector. To know more about this selector just keep reading on! 好吧，在我们开发网站或网页时，选择器是一个很常见的术语。 您可能知道很多，也可能正在实施它们。 您可能还已经注意到，所有选择器都用于选择某些元素或其他元素，这就是它们的全部用途，因此也就是名称选择器。 尽管所有选择器都具有共同的特征，但并非所有选择器都相同。 它们的行为差异很大，只有在实际实施后才能知道。 因此，既然我们正在讨论选择器，为什么不谈论一个非常具体的选择器呢？ 尽管您可能在某个时候遇到了此选择器。 我们将关注的选择器不是：first-child选择器 。 要了解有关此选择器的更多信息，请继续阅读！ Well, the name sounds a bit weird, doesn't it? Not: first child. So, that brings us to the question of what does this selector does and how is this selector different from other selectors. Well, the functioning and behavior of this selector are not very tough to understand and you can easily figure out from the name itself that for what purpose this selector is put to use. So let us look at a more formal definition of this selector so that we get a better gist of it. 好吧，这个名字听起来有些怪异，不是吗？ 不：第一个孩子。 因此，这就引出了这个选择器的作用以及该选择器与其他选择器有何不同的问题。 好吧，这个选择器的功能和行为并不是很难理解的，您可以很容易地从名称本身中得知该选择器用于什么目的。 因此，让我们看一下这个选择器的更正式定义，以便我们更好地了解它。 Definition: 定义： The not:first-child selector as the name suggests is used to select every element which is not the first child element of it's deriving parent element. Pretty simple right? The selector is not used for choosing the first child of its parent element. This selector is usually represented as an argument and it is seen in the form of not(first-child). To help you understand this in a better way, why don't you go ahead and have a look at the syntax below, 顾名思义， not : first-child选择器用于选择不是其派生父元素的第一个子元素的每个元素。 很简单吧？ 选择器不用于选择其父元素的第一个子元素。 该选择器通常表示为一个参数，并且以not(first-child)的形式出现。 为了帮助您更好地理解这一点，为什么不继续阅读下面的语法， :not(element){ //some CSS property } Example: 例： In the above example the div element contains elements, so it selects all child elements of tag except its first-child and applies the CSS styles. 在上面的示例中，div元素包含元素，因此它将选择标记的所有子元素(第一个孩子除外)并应用CSS样式。 Piece of advice: 一点建议： Now, it is time to make use of your new-found knowledge. But before you get on with that make sure you use this selector properly wherever required because you don't want your code to get ruined just because of a silly mistake right? Not just this one you must use every selector wisely to make your website or web page responsive and whenever in doubt, you will always have this article for your reference. Also, if you have some issues with our code and practice, we are always available to help you at . 现在，该利用您新发现的知识了。 但是在继续之前，请确保在所需的任何地方正确使用此选择器，因为您不希望仅仅因为一个愚蠢的错误而导致代码被破坏吗？ 您不仅必须明智地使用每个选择器，以使您的网站或网页具有响应能力，而且如有疑问，您将始终可以将本文作为参考。 另外，如果您对我们的代码和实践有任何疑问，请访问 ⓪⓪时可以为您提供帮助。 翻译自: This is one of the most well-known examples of authors misunderstanding how :first-child works. Introduced in CSS2, the :first-child pseudo-class represents the very first child of its parent. That's it. There's a very common misconception that it picks up whichever child element is the first to match the conditions specified by the rest of the compound selector. Due to the way selectors work (see here for an explanation), that is simply not true. Selectors level 3 introduces a :first-of-type pseudo-class, which represents the first element among siblings of its element type. This answer explains, with illustrations, the difference between :first-child and :first-of-type. However, as with :first-child, it does not look at any other conditions or attributes. In HTML, the element type is represented by the tag name. In the question, that type is p. Unfortunately, there is no similar :first-of-class pseudo-class for matching the first child element of a given class. At the time this answer was first posted, the newly published FPWD of Selectors level 4 introduced an :nth-match() pseudo-class, designed around existing selector mechanics as I mentioned in the first paragraph by adding a selector-list argument, through which you can supply the rest of the compound selector to get the desired filtering behavior. In recent years this functionality was subsumed into :nth-child() itself, with the selector list appearing as an optional second argument, to simplify things as well as averting the false impression that :nth-match() matched across the entire document (see the final note below). While we await cross-browser support (seriously, it's been nearly 10 years, and there has only been a single implementation for the last 5 of those years), one workaround that Lea Verou and I developed independently (she did it first!) is to first apply your desired styles to all your elements with that class: /* * Select all .red children of .home, including the first one, * and give them a border. */ .home > .red { border: 1px solid red; } ... then "undo" the styles for elements with the class that come after the first one, using the general sibling combinator ~ in an overriding rule: /* * Select all but the first .red child of .home, * and remove the border from the previous rule. */ .home > .red ~ .red { border: none; } Now only the first element with class="red" will have a border. Here's an illustration of how the rules are applied: .home > .red { border: 1px solid red; } .home > .red ~ .red { border: none; } blah first second third fourth No rules are applied; no border is rendered. This element does not have the class red, so it's skipped. Only the first rule is applied; a red border is rendered. This element has the class red, but it's not preceded by any elements with the class red in its parent. Thus the second rule is not applied, only the first, and the element keeps its border. Both rules are applied; no border is rendered. This element has the class red. It is also preceded by at least one other element with the class red. Thus both rules are applied, and the second border declaration overrides the first, thereby "undoing" it, so to speak. As a bonus, although it was introduced in Selectors 3, the general sibling combinator is actually pretty well-supported by IE7 and newer, unlike :first-of-type and :nth-of-type() which are only supported by IE9 onward. If you need good browser support, you're in luck. In fact, the fact that the sibling combinator is the only important component in this technique, and it has such amazing browser support, makes this technique very versatile — you can adapt it for filtering elements by other things, besides class selectors: You can use this to work around :first-of-type in IE7 and IE8, by simply supplying a type selector instead of a class selector (again, more on its incorrect usage in the question in a later section): article > p { /* Apply styles to article > p:first-of-type, which may or may not be :first-child */ } article > p ~ p { /* Undo the above styles for every subsequent article > p */ } You can filter by attribute selectors or any other simple selectors instead of classes. You can also combine this overriding technique with pseudo-elements even though pseudo-elements technically aren't simple selectors. Note that in order for this to work, you will need to know in advance what the default styles will be for your other sibling elements so you can override the first rule. Additionally, since this involves overriding rules in CSS, you can't achieve the same thing with a single selector for use with the Selectors API, or Selenium's CSS locators. On a final note, keep in mind that this answer assumes that the question is looking for any number of first child elements having a given class. There is neither a pseudo-class nor even a generic CSS solution for the nth match of a complex selector across the entire document — whether a solution exists depends heavily on the document structure. jQuery provides :eq(), :first, :last and more for this purpose, but note again that they function very differently from :nth-child() et al. Using the Selectors API, you can either use document.querySelector() to obtain the very first match: var first = document.querySelector('.home > .red'); Or use document.querySelectorAll() with an indexer to pick any specific match: var redElements = document.querySelectorAll('.home > .red'); var first = redElements[0]; var second = redElements[1]; // etc Although the .red:nth-of-type(1) solution in the original accepted answer by Philip Daubmeier works (which was originally written by Martyn but deleted since), it does not behave the way you'd expect it to. For example, if you only wanted to select the p here: ... then you can't use .red:first-of-type (equivalent to .red:nth-of-type(1)), because each element is the first (and only) one of its type (p and div respectively), so both will be matched by the selector. When the first element of a certain class is also the first of its type, the pseudo-class will work, but this happens only by coincidence. This behavior is demonstrated in Philip's answer. The moment you stick in an element of the same type before this element, the selector will fail. Taking the markup from the question: blah first second third fourth Applying a rule with .red:first-of-type will work, but once you add another p without the class: blah dummy first second third fourth ... the selector will immediately fail, because the first .red element is now the second p element.