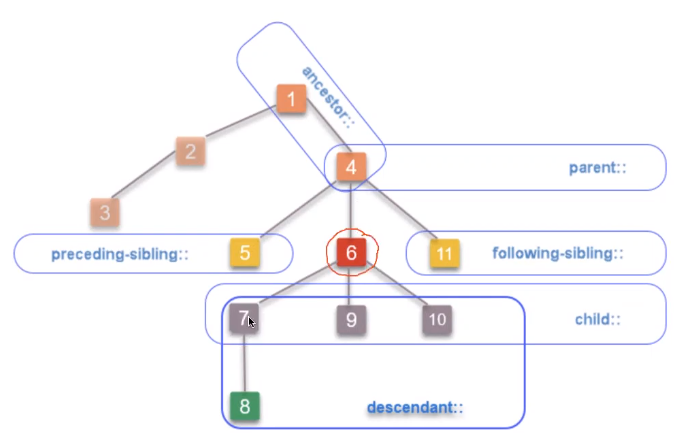
**X-path axes**

Терминология:

Это поиск элемента с помощью других, соседних, элементов (например когда нет атрибутов для зацепки)

Далее пример - нам надо найти какой то элемент без атрибутов № 6. каждая цифра это элемент. Мы можем рассматривать другие элементы в DOM вокруг 6 как ориентир, для этого надо создать терминологию какой элемент как называется по отношению к 6



6 - это **self node**, and we are identifying all other elements from 6 node.

4 - **parent tag**. (Родитель) т.е. предполагается, что 6 вложен в 4 (parent - это верхнеуровневый (на одну ступень) таг в котором находится непосредственно self node). Может быть несколько parent-ов и несколько ancestor-ов. Также 4 является parent для 5 и 11. Т.е. 5, 6, 11 вложены в 4

1 - . Это всё по отношению к 6 (а если бы мы рассматривали 4 как self node, то 1 для 4 это parent) Т.е. тег в который вложен parent self node-a называется ancestor. Другими словами parent parent-a это ancestor.

5 - **preceding sibling** (предшествующий брат), этот элемент находится сверху в DOM по отношению к self node.

11 - **following sibling** (последующий брат), этот элемент находится внизу в DOM по отношению к self node. 4, 5, 11, все являются вложенными в один общий элемент № 4 (parent) и стоят на одной ступени иерархии

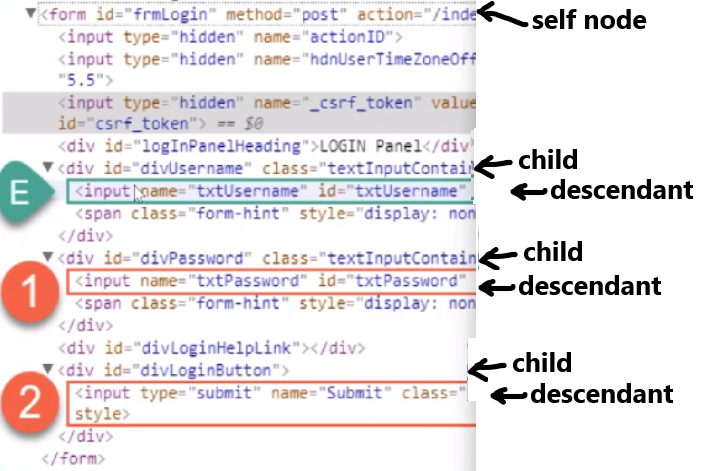
7, 9, 10 - **child**-s элементы по тношению к 6, т.е. 6 это parent для 7, 9, 10.

8 - **descendant** (потомок) для 6. Или 7,9,10 это тоже descendant для 4. А 6 descendant для 1. Другими словами child of child is descendant. Или ещё другими словами grandchild (правнук) of self node is descendant.

2 is a child for 1

3 is descendant for 1

2 and 3 are no related to **self node**

7, 9, 10, 8 - это всё following элементы

**Locating parent element**

**Синтаксис**:

**//известный\_нам\_иксПасс/parent :: название\_перента**

**//известный\_нам\_иксПасс/parent :: \*** ←если не знаешь название parent тега

**//известный\_нам\_иксПасс/ancesstor :: \*** ← так найдёт ансестора, но ансесторов будет несколько

**//известный\_нам\_иксПасс/.. ←** это сокращение, этот синтаксис тоже найдёт **перента**

Далее пример из DOM:

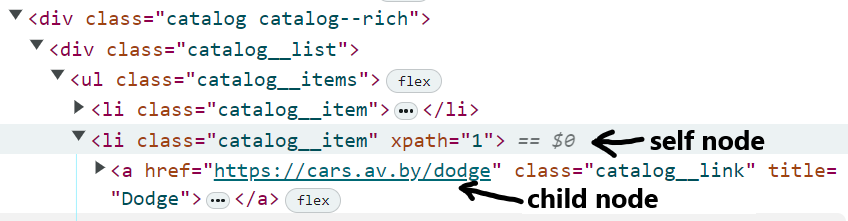


1)-нам надо найти элемент ‘a’, представим что у него нет атрибутов, мы можем взять его child элемент <span>,

тогда икс пасс будет:

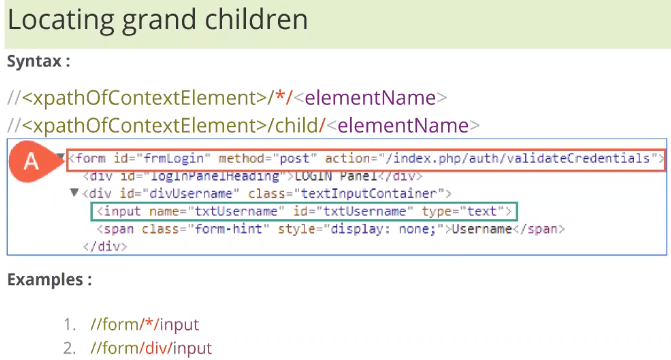
**//span[contains(text(),'Dodge')]/parent::a**

**Locating child element**



**//li[@class='catalog\_\_item']/child::a**

**//li[@class='catalog\_\_item']/a**  ← тоже самое но сокращённо, без (child::)

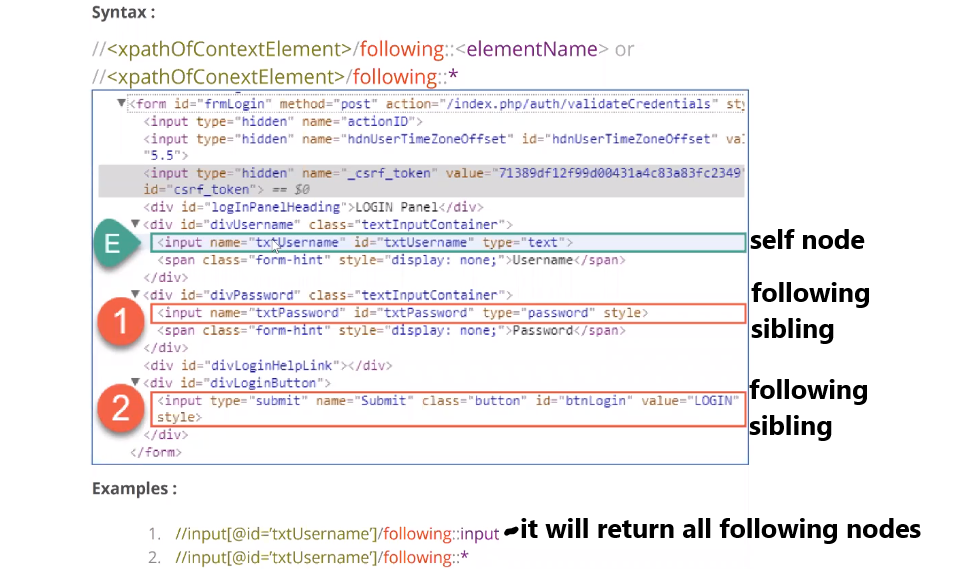


grand children it is in other words - **descendent**

с ancestor - тоже самое, такой же синтекс, смотри parent

**Locating following elements (not siblings)**

Following elements это элементы расположенные ниже в DOM от self node. На картинке ниже есть ошибка, вместо following sibling должно быть просто following.



**//input[@id = ‘txtUserName’]/following::input** - найдёт все input-ы которые идут ниже self node

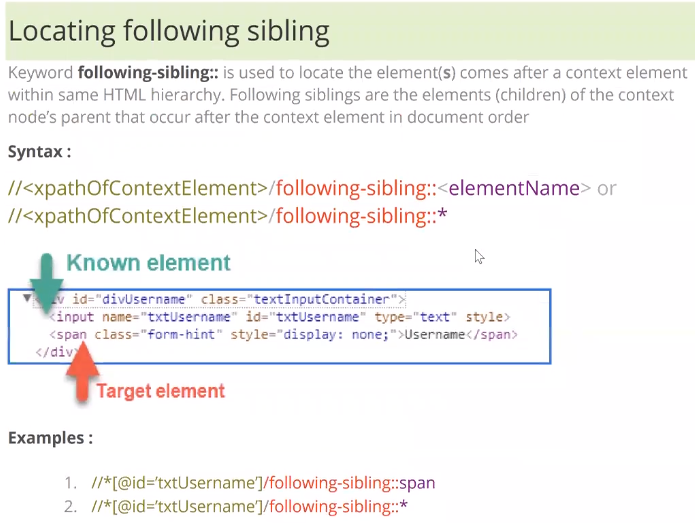
**//input[@id = ‘txtUserName’]/following::input[2]** only a specific element

**//input[@id = ‘txtUserName’]/following::\*** it will return all following nodes.

**For preceding elements sintex is the same**

**//input[@id = ‘txtUserName’]/preceding::input -** найдёт все input-ы которые идут выше self node

**//input[@id = ‘txtUserName’]/preceding::\*** - найдёт все элементы какие есть и которые идут выше self node



На примере выше //input[@name=’txUserName’] - self node, а span который идёт сразу ниже является following sibling.

Following siblings - это элементы которые находятся на одном уровне иерархии в DOM с self node, но расположены ниже

//\*[@id = ‘txtUsername’]/following-sibling::span ← вернёт все span-ы

//\*[@id = txtUsername’]/following-sibling::\* ← вернёт все following sibling элементы

**PRECEDING SIBLINGS**

Всё тоже как и с following sibling, только прописываешь preceding-sibling

//\*[@id = ‘txtUsername’]/preceding-sibling::span

//\*[@id = txtUsername’]/preceding-sibling::\*