Assume you are expert in finding xpath and a good guide to explain to beginner to advance level then provide all the examples to find xpath like parent seach and also with how the html code looks like

Great! Let's dive deep into **XPath** from **beginner to advanced level** with **real HTML code examples** and explanations.

🔰 What is XPath?

XPath (XML Path Language) is used to **navigate through elements and attributes in XML/HTML documents**. It's widely used in **automation tools like Selenium, Cypress, and Playwright** to locate web elements.

✅ Basic Syntax

//tagname[@attribute='value']

Example:

//input[@id='username']

🌱 Beginner Level

---

1. Absolute XPath

Finds the element using the **full path** from the root.

- **HTML**:

<html>

```
 <body>
  <div>
    <input type="text" id="username"/>
  </div>
 </body>
</html>
```

- **XPath**:

```
//html/body/div/input
```

🚫 **Not recommended**: Changes in DOM can break the path.

---

2. Relative XPath

Starts from anywhere using `//`.

- **HTML**:

```
<input type="text" id="username"/>
```

- **XPath**:

```
//input[@id='username']
```

✅ More stable and commonly used.

---

3. Using Multiple Attributes

```
//input[@id='username' and @type='text']
```

---

4. Using `contains()`

Partial match of attribute values.

- **HTML**:

<input type="text" id="user-name-text"/>

- **XPath**:

//input[contains(@id, 'user')]

---

5. Using `starts-with()`

//input[starts-with(@id, 'user')]

---

6. Using `text()`

- **HTML**:

<button>Submit</button>

- **XPath**:

//button[text()='Submit']

🌿 Intermediate Level

---

7. Using `parent` Axis

- **HTML**:

  <label>Username</label>
  <input id="uname" type="text"/>

- **XPath to go from input to parent `div`**:

//input[@id='uname']/parent::div

---

8. Using `child` Axis

- **HTML**:

```
<div id="form">
  <input id="email"/>
</div>
```

- **XPath**:

```
//div[@id='form']/child::input
```

---

9. Using `ancestor` Axis

- **HTML**:

```
<div id="outer">
  <div>
    <input id="searchBox"/>
  </div>
</div>
```

- **XPath to get `div#outer` from `input`**:

```
//input[@id='searchBox']/ancestor::div[@id='outer']
```

---

10. Using `following` and `preceding`

- **HTML**:

```
<h1>Header</h1>
<p>First paragraph</p>
<p>Second paragraph</p>
```

- **XPath (get 2nd p after h1)**:

```
//h1/following::p[2]
```

---

11. Using `following-sibling`

- **HTML**:

```
<div>
  <label>Email</label>
  <input type="text"/>
</div>
```

- **XPath**:

```
//label[text()='Email']/following-sibling::input
```

---

12. Using `preceding-sibling`

- **HTML**:

```
<div>
  <label>Email</label>
  <input type="text"/>
</div>
```

- **XPath**:

```
//input[@type='text']/preceding-sibling::label
```

🌳 Advanced Level

---

13. Indexing Elements

- **HTML**:

```
<ul>
  <li>Home</li>
  <li>About</li>
  <li>Contact</li>
</ul>
```

- **XPath for second item**:

```
//ul/li[2]
```

---

14. Using `last()`

- **XPath to get last list item**:

```
//ul/li[last()]
```

---

15. XPath with `and` / `or`

```
//input[@type='text' and @name='email']
```

```
//input[@type='text' or @type='password']
```

---

16. Wildcard `*`

- **Any tag with id 'login'**:

```
//*[@id='login']
```

---

17. Using `normalize-space()`

Useful to trim whitespaces.

- **HTML**:

<button> Login </button>

- **XPath**:

//button[normalize-space(text())='Login']

---

18. Using Grouping ( ) in XPath

- **HTML**:

Save
Submit

- **XPath**:

//div[(text()='Save' or text()='Submit') and @class='btn']

---

19. Chained XPath

- **HTML**:

<table>
  <tr>
    <td>Name</td>
    <td>John</td>
  </tr>
</table>

- **XPath for value "John"**:

//td[text()='Name']/following-sibling::td

---

20. Custom XPath for Dynamic IDs

- **HTML**:

<input id="user_12345" />

🔸 **XPath**:

//input[starts-with(@id, 'user_')]

🎁 Bonus Tip: Test XPath in Browser

🔹 Press `F12` → `Console` → Use:

$x("//input[@id='username']")

Would you like a downloadable **PDF or code snippet library** with these examples?