```
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=ISO-8859-1">
<title>Insert title here</title>
</head>
<body>
        <table border="1">
            <tr><th colspan="4">HISTORY</th></tr>
            <tr> <td>Title</td> <td>Author</td>     <td>Year</td>   <td>Price</td>  </tr>
            <tr>
                <td><div name="manish" surname="joshi" id="vs">Veer Shivaji</div></td>  <td>Babasaheb Purandare</td><td>1988</td>
                <td>420 INR</td>
            </tr>
        </table>
        <table border="1">
        <table border="1">
        <table border="1">
        <div><a href="#">click</a></div>
</body>
</html>
```

.	Represent the Current Node e.g.
	`String var=driver.findElement(By.id("vs")).findElement(By.xpath(".")).getText();`
	`System.out.println(var);`

..	Represent the Parent Node of the Selected Node. e.g. `driver.findElement(By.id("vs")).findElement(By.xpath("..")).getTagName();`
/	Always Starts From the Root Node. i.e. HTML node. Always search the element from the Beginning. e.g. driver.FindElement("/html").getAttribute("outerHTML")

//	Start finding out the Element into given web page. It may be present at any level. E.g.
	`String var=driver.findElement(By.id("vs")).findElement(By.xpath("..")).findElement(By.xpath("//div")).getText();`
	`System.out.println(var);`

@	@ Symbol we used only for Attributes. In Selenium we identify the elements, so here we can take the help of @ tag to identify the elements with the help of matching attributes.
	Example 1:
	
	`String var=driver.findElement(By.xpath("//div[@surname]")).getAttribute("outerHTML");`
	Here we are finding out an element having tag name div and having attribute name as "surname"
	
	Note: In Above example we are identifying the Element having Tag Name div and which contains attribute name "surname"
	
	Example 2:
	
	`List<WebElement> matchingObjects=driver.findElements(By.xpath("//div[@name][@surname]"))`

	;
	for(WebElementsingwebelement : matchingObjects)
	{
	
	System.*out*.println(singwebelement.getAttribute("name")+"
	"+singwebelement.getAttribute("surname"));
	
	}
	
	Note: In Above example we are finding out the Element having Tag Name div and which contains attribute name "surname" and "name"
	
	Example 3:
	
	List<WebElement>
	matchingObjects=*driver*.findElements(By.*xpath*("//div[@name='manish'][@surname='joshi']"));//.getAttribute("outerHTML");
	for(WebElementsingwebelement : matchingObjects)
	{
	System.*out*.println("--------
	"+singwebelement.getAttribute("name")+"
	"+singwebelement.getAttribute("surname"));
	
	}
	
	Note: In Above example we are finding out the Element having Tag Name div and which contains attribute name "surname" and "name" having values 'manish' and 'joshi' respectively.

Tagname [index]	List<WebElement> matchingObjects=*driver*.findElements(By.*xpath*("//table/table[0]")); Note: In above example it will first find out the element having tag name table and inside this table tag he will find out the another element having tag name table having index 0.
Tagname [last()]	List<WebElement> matchingObjects=*driver*.findElements(By.*xpath*("//table/table[last()]")); Note: In above example it will first find out the element having tag name table and inside this table tag he will find out the child elements having tag name Table, but will return you the last child object reference.
Tagname [last()-1]	List<WebElement> matchingObjects=*driver*.findElements(By.*xpath*("//table/table[last()-1]")); Note: In above example it will first find out the element having tag name table and inside this table tag he will find out the child elements having tag name Table, but will return you the second last child object reference.
Tagname [position ()<3]	List<WebElement> matchingObjects=*driver*.findElements(By.*xpath*("//table/table[position()<3]")); Note: In above example it will first find out the element having tag name table and inside this table tag he will find out all the child elements having tag name Table, but will return you the first 2.
	List<WebElement>matchingObjects=*driver*.findElements(By.*xpath*(" //table//*[th=\"COOKING\"]"))

	Above Example will return you all the child elements under table whose th tag matches with cooking
|	OR
//book/title | //book/price	Selects all the title AND price elements of all book elements
*	Matches any Element
//*	Selects all elements in the document
Contains	Finds the Element under table whose th element contains "CHIL" //table/*[contains(.,"CHIL")]
Case Insensitivity	//table//*[contains(translate(.,'ABCDEFGHIJKLMNOPQRSTUVWXYZ','abcdefghijklmnopqrstuvwxyz'),'cooking')] Identifying child element which contains cooking text. Here "." represents the current element.