```
== XPath Summary ==

Build the document tree: doc function
- fn:doc("quiz.xml")

Go parent to an element child: /
- fn:doc("quiz.xml")/quiz/questions/mc-question/question

Go one level down without caring what the element name is: *
- fn:doc("quiz.xml")/*/*/*/@qid

Find any way to an element or attribute, in any descendant: //
- fn:doc("quiz.xml")//mc-question/@qid
- fn:doc("quiz.xml")/quiz/class-responses//@qid
- fn:doc("quiz.xml")//questions//@qid

Index into a node set: [#]
- fn:doc("quiz.xml")//class-responses/student[2]

Go to an attribute: @att-name
- fn:doc("quiz.xml")/quiz/questions/mc-question/@solution

Select results that satisfy an existentially quantified condition: [expression]
- use subelementName to refer to a subelement of the current element
        fn:doc("races.xml")/races/race[result <= 3.50]
- use @att-name to refer to the value of an attribute of the current element
        fn:doc("races.xml")/races/*[@name > "K"]
- use "." to refer to the entirety of the current element
        fn:doc("races.xml")/races/race/result[. <= 3.50]

Functions (there are tons more!)
- Get the "guts" of an element: text()
        fn:doc("quiz.xml")/quiz/questions/mc-question/question/text()
- Aggregate: max(), min(), avg(), count()
        These can end the path expression, or be inside a condition
```