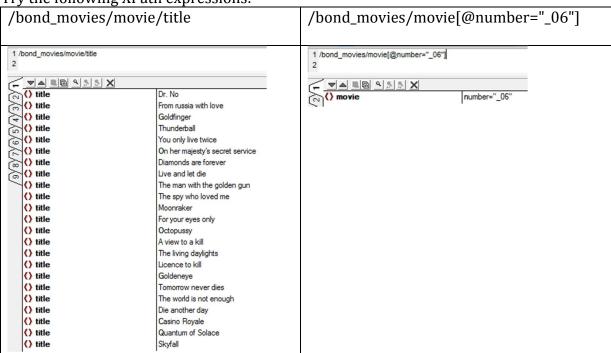## 3 - XPath | Exercise 1: Addressing Elements & Attributes

1.  Open the XPath 2.0 Online Tester and copy-paste the file bond_movies.xml to the text field.

    ➔ *XMLSpy*

▾ XML
▴ bond_movies

| | = month | August | | | | | |
|---|---|---|---|---|---|---|---|
| | = year | 2013 | | | | | |
| ▴ movie (23) | | | | | | | |

| | = number | () title | () bond | () bond_girl | () regie | () year | () duration |
|---|---|---|---|---|---|---|---|
| 1 | _01 | Dr. No | Sean Connery | Ursula Andress | Terence Young | 1962 | 105 |
| 2 | _02 | From russia with love | Sean Connery | Daniela Bianchi | Terence Young | 1963 | 110 |
| 3 | _03 | Goldfinger | Sean Connery | Honor Blackman | Guy Hamilton | 1964 | 106 |
| 4 | _04 | Thunderball | Sean Connery | Claudine Auger | Terence Young | 1965 | 125 |
| 5 | _05 | You only live twice | Sean Connery | Mie Hama | Lewis Gilbert | 1967 | 112 |
| 6 | _06 | On her majesty's secret service | George Lazenby | Diana Rigg | Peter R. Hunt | 1969 | 136 |
| 7 | _07 | Diamonds are forever | Sean Connery | Jill St. John | Guy Hamilton | 1971 | 120 |
| 8 | _08 | Live and let die | Roger Moore | Jane Seymour | Guy Hamilton | 1973 | 116 |
| 9 | _09 | The man with the golden gun | Roger Moore | Britt Ekland | Guy Hamilton | 1975 | 120 |
| 10 | _10 | The spy who loved me | Roger Moore | Barbara Bach | Lewis Gilbert | 1977 | 120 |
| 11 | _11 | Moonraker | Roger Moore | Lois Chiles | Lewis Gilbert | 1979 | 122 |
| 12 | _12 | For your eyes only | Roger Moore | Carole Bouqet | John Glen | 1981 | 121 |

2.  Try the following XPath expressions:

| /bond_movies/movie/title | /bond_movies/movie[@number="_06"] |
|---|---|
| 1 /bond_movies/movie/title<br>2 | 1 /bond_movies/movie[@number="_06"]<br>2 |
| () title — Dr. No<br>() title — From russia with love<br>() title — Goldfinger<br>() title — Thunderball<br>() title — You only live twice<br>() title — On her majesty's secret service<br>() title — Diamonds are forever<br>() title — Live and let die<br>() title — The man with the golden gun<br>() title — The spy who loved me<br>() title — Moonraker<br>() title — For your eyes only<br>() title — Octopussy<br>() title — A view to a kill<br>() title — The living daylights<br>() title — Licence to kill<br>() title — Goldeneye<br>() title — Tomorrow never dies<br>() title — The world is not enough<br>() title — Die another day<br>() title — Casino Royale<br>() title — Quantum of Solace<br>() title — Skyfall | () movie — number="_06" |

3.  Give an XPath expression that outputs:
    Halle Berry
    `/bond_movies/movie[@number="_20"]/bond_girl`

4.  Use the XPath function text() for accessing the content of a text node.
    */bond_movies/movie[@number="_20"]/bond_girl/text()*

    */bond_movies/movie[@number="_20"]/title/text()*

## 3 - XPath | Exercise 2

1.   Make a prediction what the following expression returns and verify
     /descendant::title[text() = "Dr. No"]/../regie
     *<regie>Terence Young</regie>*


     Search XPath expressions for the following queries:
2.   Which number has the Bond movie with Maud Adams ?
     */descendant::bond_girl[text() = "Maud Adams"]/../@number*


3.   How many times (use `count()`) did Sean Connery play Bond ?
     *count(/descendant::bond[text() = "Sean Connery"])*
     → *6*


4.   My friends and I plan a James Bond movie night where we want to watch all movies
     from the very first to the very last. How much time (use `sum()`) will this take ?
     *sum(/descendant::duration)*
     → *2804*


5.   List all movies over 120min.
     */bond_movies/movie/duration[number() >120]/../title*
     ```
     title Thunderball
     title On her majesty's secret service
     title Moonraker
     title For your eyes only
     title Octopussy
     title A view to a kill
     title The living daylights
     title Licence to kill
     title Goldeneye
     title The world is not enough
     title Die another day
     title Casino Royale
     title Skyfall
     ```


6.   List all Bond actors without double naming (use distinct-values())
     *distinct-values(/descendant::bond)*


7.   Challenge your neighbour and find a really complex XQuery task that you expect him /
     her not to be able to solve. Treat your neighbour to a cup of coffee if you lost.
     *Alle Filme von John Glen (Regie) zusammen mit Roger Moore (Bond) ohne Octopussy (Film)*
     */descendant::regie[text() = "John Glen"]/../bond[text()="Roger Moore"]/../title[text() != "Octopussy"]*
     ```
     title For your eyes only
     title A view to a kill
     ```

     */descendant::movie[regie = "John Glen" and bond = "Roger Moore" and title != "Octopussy"]*


8.   Has it ever happened that two Bond movies were produced with the same director?
     Formulate the query.
     *for $i in distinct-values(/descendant::regie)*
       *return concat($i, " produced ", count(/bond_movies/movie[regie = $i]), " moves.")*

# 3 - XPath | Exercise 3

1.  Formulate an XPath query that produces the following output.

    Terence Young produced 3 movies.

    Guy Hamilton produced 4 movies.

    Lewis Gilbert produced 3 movies.

    Peter R. Hunt produced 1 movies.

    John Glen produced 5 movies.

    Martin Campbell produced 2 movies.

    Roger Spottiswoode produced 1 movies.

    Michael Apted produced 1 movies.

    Lee Tamahori produced 1 movies.

    Marc Forster produced 1 movies.

    Sam Mendes produced 1 movies.

```
for $i in distinct-values(/descendant::regie)
  return concat($i, " produced ", count(/bond_movies/movie[regie = $i]), " movies.")
```

2.  Modify the query to show only successful producers (> 1 movie).

    Terence Young produced 3 movies.

    Guy Hamilton produced 4 movies.

    Lewis Gilbert produced 3 movies.

    John Glen produced 5 movies.

    Martin Campbell produced 2 movies.

```
for $i in distinct-values(/descendant::regie) return
  if (count(/bond_movies/movie[regie = $i]) gt 1) then
    concat($i, " produced ", count(/bond_movies/movie[regie = $i]), " movies.")
  else ""
```