

**DIPLOMA IN COMPUTER SCIENCE**

**SCHOOL OF COMPUTING**

|  |
| --- |
| **Assessment: Assignment** |

|  |  |
| --- | --- |
| **Course Code** | **DCS 3003** |
| **Course Name** | **Website Design** |
| **Academic Session** | **September 2023** |
| **Lecturer name** | **Ms Teo Choon Yeh** |

|  |  |
| --- | --- |
| **Released Date:** | **15 September 2023** |
| **Submission**  **Due Date:** | **5 November 2023** |

|  |  |  |  |
| --- | --- | --- | --- |
| **Weightage (%):** | **15** | **Mark Obtained** | **15** |

**Student’s Declaration**

I certify that this assignment is my own work, and appropriately acknowledged wherever material adapted from other sources. I understand that plagiarism, cheating, collusion, fraud, fabrication, or falsification of data are not acceptable. I agree that if at any time it is shown that I have significantly misrepresented material submitted to Crescendo International College, any marks or credits awarded to me based on that material may be revoked.

|  |  |  |  |
| --- | --- | --- | --- |
| Student Name | Ferdinand Gan | Student ID | DCS2309267 |
| Signature |  | Date | 8/10/2023 |

Table of Contents

1. [Introduction 5](#_Toc149864793)
2. [Content 6](#_Toc149864794)

[What is XML? 6](#_Toc149864795)

[Impact of XML 7](#_Toc149864796)

[Advantages and disadvantages of XML 8](#_Toc149864797)

[Pros 8](#_Toc149864798)

[Cons 10](#_Toc149864799)

[Similarities between XML and HTML 12](#_Toc149864800)

[Difference between XML and HTML 13](#_Toc149864801)

1. [Conclusion 14](#_Toc149864802)
2. [Reference Lists 16](#_Toc149864803)
3. [Assignment 19](#_Toc149864804)

[Picture of the web page 19](#_Toc149864805)

[Explanation 20](#_Toc149864806)

[The function 20](#_Toc149864807)

[Advantages of XML used 21](#_Toc149864808)

[Coding 22](#_Toc149864809)

[1. HTML file code 22](#_Toc149864810)

[2. XML file code 27](#_Toc149864811)

# Introduction

Have you ever heard of XML? Which stands for eXtensible Markup Language, or XML, was initially presented by the World Wide Web Consortium (W3C) in the late 1990s and has since grown to be an essential internet technology. Renowned for its adaptable and widely utilized markup language, which aims to transmit and exchange data in an organized manner between various computer systems, including websites, databases, and third-party apps, as well as organize and store various data types, which comprises numbers, strings, and executable code. This markup language permits information to flow between various computer systems smoothly, notably databases, websites, and outside applications. What makes it remarkable is the fact that the syntax it uses is both machine- and human-readable, which renders simplicity for both software and people to comprehend. Furthermore, it is is vital for database administration and web development because it provides a strong framework for designing unambiguous data structures through the use of elements, attributes, and hierarchical connections as its building blocks.

# Content

## What is XML?

XML (Extensible Markup Language) is a markup language that offers a way to arrange and convey data in a manner that is both machine and human-readable. Widely utilized for keeping and retrieving information in numerous databases along with web development.

It is a flexible language that may be used in multiple scenarios regarding data access, transfer, and storage; yet, it is not a programming language. Its self-documenting format, which specifies the structure, field names, and specific values, enhances understanding.

An XML document's constituent pieces, known as elements, are determined by the tags that enclose its contents. A start tag, an end tag, and the content placed between the tags compose each element.

## Impact of XML

XML (Extensible Markup Language) has had a significant impact on web development. It is frequently used in web services to provide internet-based communication between many applications. It improves communication between many applications by assisting to specify the arrangement and content of web services.

For instance, data exchange and storage in web applications or databases. Web developers use XML for storing and retrieving structured data from several databases, as well as to specify the format and structure of data that is transmitted across various applications. Likewise it is a flexible language which renders more data accessible through permitting access to information gathered from XML data sources from many applications. And because of its platform-independent nature, it also offers a simplification of data sharing and transportation between various systems without the need of any conversion. Oracle and Microsoft SQL Server are two instances.

With its potential to increase the availability of database outputs across diverse systems and make data transfer between different systems or multiple organizations by the option of "Export to XML," not only does it make it easier for different applications to access and use the data but also improve data interchange and enables cross-platform development which can run on different platforms and systems. Thanks to its self-documenting format that describes the structure and field names as well as specific values, it makes it easier to comprehend and work with. In response to the increased demand for XML documents produced by business transactions, several IT professionals throughout a range of industries have been searching for ways to effectively analyze and handle XML data.

In summary, XML has made a significant change on both web development and data management in different industries by simplifying data sharing and transport, increasing data availability, and providing a versatile and flexible markup language for data exchange, storage, and presentation on the web. Followed by its availability of database outputs and its improvement on data interchange, storage and retrieval which enables cross-platform development.

## Advantages and disadvantages of XML

### Pros

With its extraordinary adaptability and broad use in a variety of fields. Its independence from platforms and languages serves as one of its primary benefits. It is the preferred option for data interchange in an array of distinct and heterogeneous circumstances since it functions on major operating systems and interacts effortlessly with any programming language.

On top of that, data management is made easier by XML's human-readable format. Given that XML documents are composed in plain language, troubleshooting, manual editing, and data examination are made more facile for humans to execute. This transparency pledges that handling XML data will never cease to be straightforward and readily accessible.

XML's ability to portray structured data is another powerful feature. The construction of distinct data structures is rendered possible by XML with the incorporation of elements, attributes, and hierarchical relationships. This capacity is especially helpful for successfully describing complicated datasets.

Besides, users can specify their own tags and create unique document structures as a result of XML's extensibility. Because of its flexibility, XML can be used in a variety of contexts and sectors without being restricted by preset components.

When it pertains to interoperability, XML shines, facilitating data interchange across various platforms and systems. In light of this capacity, XML is guaranteed to be supported by all platforms, facilitating seamless communication across different programs and databases, independent of the technologies that underlie them.

Plus, XML facilitates validation against preset schemas (such as DTD or XSD), guaranteeing compliance with particular structural guidelines. Maintaining data accuracy and reliability is vital for data management, and this validation method helps to achieve this.

XML's prowess in data transformation cannot be understated. Technologies like XSLT enable the conversion of XML data into different formats, such as HTML, for display in web browsers. This revolutionary capability increases XML's resilience to a broad spectrum of output requirements.

Moreover, it allows for the creation of customizable document structures tailored to specific domains or applications. This flexibility gives users the ability to express data in a way that best suits their individual requirements and makes the most sense.

XML integrates with databases, serving as a format for storing and exchanging data. Integration with current data management systems is ensured via the built-in XML support incorporated in many database systems.

Additionally, it accommodates the inclusion of metadata, offering a means to provide supplementary information about the data. This feature strengthens the information's overall understanding by helping to document and classify data pieces.

Finally, XML is widely used in many different fields and companies. Web services, data transfer, configuration files, scientific data representation, financial transactions, along with other fields all use it as an accepted convention. This widespread usage demonstrates XML's continuous appeal and vital function in many data-related applications.

To sum it off, XML's adaptability, structured representation, interoperability, validation support, and broad industry adoption establish it as a fundamental technology for effective data management and interchange across diverse applications. Its ongoing applicability demonstrates its lasting worth in the always changing field of information technology.

### Cons

XML, while widely utilized in web and application development, does come with its set of drawbacks. A significant drawback is that, in comparison to other formats like JSON, it possesses a verbose syntax that is often more redundant. Larger file sizes as a result of this verbosity could impact storage needs and network capacity. In addition, redundancy may be introduced by XML's obligatory opening and closing tags for each item of data, making documents harder to read and maintain.

Moreover, parsing XML documents can be computationally expensive, especially for large and complex files. Applications that primarily depend on regular parsing of XML data might encounter performance issues as an outcome of this processing overhead. Furthermore, XML faces limitations in handling binary data efficiently, unlike other formats which may have more specialized support for such data types.

Another consideration is the representation of relationships between data elements. XML may not be as intuitive as other data models, particularly in scenarios involving complex parent-child relationships. Furthermore, maintaining XML schemas can be challenging, especially since they contain complicated data structures, which contributes labor to the establishment and upkeep of schemas.

While XPath and XQuery exist for querying XML documents, they may not be as expressive or powerful as some other query languages, potentially making complex data extraction tasks more challenging. Additionally, some databases might not interact with XML naturally, necessitating extra setups or libraries for efficient handling.

In situations where extremely high performance and low latency are paramount, the parsing and processing overhead of XML could be a limiting factor, making it less suitable for real-time systems. Meanwhile, erroneous handling of XML data is susceptible to security flaws like XML injection attacks, underscoring the significance of appropriate validation and sanitization.

For simple data structures, XML might introduce unnecessary complexity. In such cases, a more lightweight format like JSON could be a more suitable choice. The particular needs and limitations of the application or system concerned issue should ultimately choose the data format to use, taking into account the advantages and disadvantages of each format.

## Similarities between XML and HTML

Both XML and HTML can work together in several ways, and are both globally used in web development and application development. Processing on various platforms and are compatible with different web browsers and applications. But together it can create powerful web experiences.

They are markup languages that provide the structure for data, and use tags to identify elements within the data which can be parsed by an application and read by software and browsers. Following a tree-like parsing model where elements are hierarchically organized. The visualization and interpretation of the content is rendered easier by this framework.

Supporting the use of attributes within elements which provides additional information about an element and use of entities, which are placeholders for special characters. Allowing inclusion of text content within elements.

## Difference between XML and HTML

Fundamentally, XML is a text-based standard that defines the meaning and structure of data through tags. These tags can be changed based on the data being conveyed and are akin to those found in HTML (HyperText Markup Language). Because of its extensibilities, it is an infrastructure for defining markup languages that the user can define.. However, it is case-sensitive and more strict in formatting with the XML tags closed. Apart from that, it is dynamic in nature and content-driven. Mostly used to represent data in a hierarchical structure and exchange data between different systems.

On the contrary, HTML is a predefined markup language that is used to create websites and display material on them in web browsers. It is typically used for web development. It has predefined tags for various elements like headings, paragraphs, lists, links, and more for specific purposes like headings, paragraphs, tables, forms, etc. Both static in nature and presentation-driven. Particularly, there is no need to close the tags, and it is not case-sensitive. Offers more flexibility in formatting and will attempt to render incorrectly formatted code.

Overall, XML focuses solely on data representation and does not define how the data should be presented or styled. It's usually combined with other technologies (like XSLT) for transforming XML into other formats like HTML for display. Whereas, HTML features properties like class and style that enable components to be directly styled through CSS (Cascading Style Sheets) to alter how the content is presented visually.

# Conclusion

In conclusion, XML, or Extensible Markup Language, has emerged as a foundational technology that has revolutionized data organization, storage, and exchange on the internet. Given its accommodating and prevalent markup language, a wide range of data types can be represented in an organized manner, facilitating smooth communication across disparate computer systems. The implications of XML on data management and web development are significant.

XML's defining feature is its structured format, composed of elements, attributes, and content. In spite of guaranteeing human-readability, this self-documenting method offers a structured format for data representation. In light of this feature, web development and database management benefit greatly from it, since it facilitates efficient internet-based application communication.

The platform and language independence of XML further enhance its versatility, allowing it to integrate seamlessly into heterogeneous environments. Thanks to its versatility and customization options, users may establish their own tags, which makes it appropriate for a broad range of industries and applications. Furthermore, XML excels at interoperability, guaranteeing that it is widely accepted and capable of enabling seamless communication across many platforms and systems.

However, XML is not without its challenges, to increased storage, transmission, and processing costs. Moreover, it might take more work to articulate overlapping node relationships in its hierarchical form. XML data structure design can get more difficult due to namespace support and the difference between content and attributes.

Despite these drawbacks, XML remains a pivotal technology in web and application development. Its structured representation, interoperability, validation support, and industry-wide adoption establish it as a fundamental tool for effective data management and interchange. The choice between XML and other formats should be made judiciously, considering the specific requirements and constraints of the application. Ultimately, XML's enduring relevance and continued use in diverse applications speak to its enduring value in the ever-evolving landscape of information technology.

# Github links

# Reference Lists

1. *What is XML?* [Online]. Available at: <https://aws.amazon.com/what-is/xml/>
2. *XML introduction.* [Online]. Available at: <https://www.w3schools.com/xml/xml_whatis.asp>
3. *Advantages of XML.* [Online]. Available at: <https://www.ibm.com/docs/en/i/7.3?topic=introduction-advantages-xml>
4. *Explaining XML.* [Online]. Available at: <https://hbr.org/2000/07/explaining-xml>
5. *What is XML?* [Online]. Available at: <https://www.techtarget.com/whatis/definition/XML-Extensible-Markup-Language>
6. *What is XML?* [Online]. Available at: <https://builtin.com/data-science/xml>
7. *Similarities between XML and HTML.* [Online]. Available at: <https://www.steadyrun.com/similarities-xml-html>
8. *XML vs HTML: Differences and similarities.* [Online]. Available at: <https://venturebeat.com/datadecisionmakers/xml-vs-html-differences-and-similarities/>
9. *Exploring the Differences Between HTML and XML.* [Online]. Available at: <https://blog.hubspot.com/website/xml-vs-html>
10. *HTML vs XML.* [Online]. Available at: <https://www.geeksforgeeks.org/html-vs-xml/>
11. *XML vs HTML: Learn the Difference Between These Markup Languages.* [Online]. Available at: <https://kinsta.com/blog/xml-vs-html/>
12. *Explaining XML by Eileen Roche.* [Online]. Available at: <https://hbr.org/2000/07/explaining-xml>
13. *Difference Between HTML and XML.* [Online]. Available at: <https://www.knowledgehut.com/blog/web-development/html-vs-xml>
14. *XML Vs. HTML: What is Difference Between XML and HTML?* [Online]. Available at: <https://www.guru99.com/xml-vs-html-difference.html>
15. *What’s the Difference Between HTML and XML?* [Online]. Available at: <https://aws.amazon.com/compare/the-difference-between-html-and-xml/>
16. *Difference Between XML and HTML.* [Online]. Available at: <https://byjus.com/free-ias-prep/difference-between-xml-and-html/>
17. *Advantages of XML.* [Online]. Available at: <https://www.ibm.com/docs/en/i/7.3?topic=introduction-advantages-xml>
18. *Advantages and Disadvantages of XML.* [Online]. Available at: <https://beginnersbook.com/2018/10/advantages-and-disadvantages-of-xml/>
19. *Features and Advantages of XML.* [Online]. Available at: <https://www.javatpoint.com/features-and-advantages-of-xml>
20. *Advantages and Disadvantages of XML.* [Online]. Available at: <https://www.spec-india.com/blog/advantages-and-disadvantages-of-xml>
21. *Advantages and Disadvantages of XML. [*Online]. Available at: <https://www.rfwireless-world.com/Terminology/Advantages-and-Disadvantages-of-XML.html>
22. *XML Module Conclusion.* [Online]. Available at: <https://www.javadeploy.com/xml-programming/module2/xml-module-conclusion.jsp#google_vignette>
23. *Conclusion of XML.* [Online]. Available at: <https://www.oreilly.com/library/view/strategicxml/0672321750/0672321750_ch13lev1sec6.html>
24. *Introduction to XML.* [Online]. Available at: <https://intellipaat.com/blog/what-is-xml/>
25. *The Role of XML in Modern Web Development.* [Online]. Available at: <https://www.hurix.com/the-role-of-xml-in-modern-web-development/>
26. *What Is XML Used For?* [Online]. Available at: <https://www.codecademy.com/resources/blog/what-is-xml-used-for/>
27. *5 Basic Reasons You Should Use XML.* [Online]. Available at: <https://www.thoughtco.com/reasons-to-use-xml-3471386>
28. *The Impact of XML on Databases and Data Sharing.* [Online]. Available at: <https://www.researchgate.net/publication/2452742_The_Impact_of_XML_on_Databases_and_Data_Sharing>
29. *XML's Impact on Databases and Data Sharing.* [Online]. Available at: <https://www.researchgate.net/publication/220478215_XML's_Impact_on_Databases_and_Data_Sharing>
30. *An empirical study of XML data management in business information systems.* [Online]. Available at: <https://www.sciencedirect.com/science/article/abs/pii/S016412120500155X>
31. *Is XML the way for corporate data?* [Online]. Available at: <https://www.cnet.com/tech/tech-industry/is-xml-the-way-for-corporate-data/>
32. *Extensible Markup Language (XML).* [Online]. Available at: <https://mailchimp.com/marketing-glossary/xml/>

# Assignment

## Picture of the web pageA screenshot of a computer Description automatically generated

## Explanation

### The function

1. The HTML file contains a page structure with a centered heading, a horizontal line, and a table header with columns for 'Year', 'Death Rate', and 'Growth Rate'.
2. There's an empty <div> element with the id 'container' where the dynamically generated table content will be inserted.
3. The JavaScript code uses XMLHttpRequest to fetch XML data from xml\_content1.xml, parses it, and generates table rows based on the content.
4. Each 'part' element in the XML file corresponds to a row in the table, with 'year', 'deathrate', and 'growthrate' values being displayed in the respective columns.

### Advantages of XML used

The XML file I have utilized illustrates a table of a decade of mortality data spanning multiple years. Employing XML for this type of data representation has several advantages to it.

Firstly, XML provides a structured format for organizing and categorizing information. This structured approach is very effective for organizing large datasets. Furthermore, XML is designed to be human-readable, which implies that both humans and machines can understand it. Because of this, it is a good solution for easing data interchange between various systems and applications.

Furthermore, XML is platform-independent, which means it may be used on a variety of operating systems and read by a variety of programming languages. Because of its adaptability, it can represent a wide range of data kinds and structures, allowing it to accept different types of information and adapt to changing requirements.

The extensive support for XML across various programming languages and platforms further contributes to its popularity. Because of this interoperability, it is a good solution for data sharing between systems that may not be using the same technological stack. Furthermore, XML is a well-established data representation and exchange standard, with a solid ecosystem of tools and libraries for parsing and working with XML documents.

Another advantage of XML lies in its validation capabilities. XML documents can be validated against a schema or Document Type Definition (DTD), ensuring they adhere to a specific structure and meet predefined criteria. This validation process adds an extra layer of reliability to the data.

XML also interfaces seamlessly with web technologies like JavaScript, allowing for dynamic data processing and display in web applications. Its compatibility with these technologies broadens its applicability in modern software development.

Lastly, XML can be a dependable format for long-term data preservation. It is not bound to any one technology or platform, assuring long-term data longevity. Because of its endurance, it is a reliable alternative for saving information for future reference.

In conclusion, the decision to use XML depends on the specific requirements of your application and the nature of the data at hand. In your case, XML proves to be a fitting choice for representing mortality data in a structured, standardized, and easily accessible format. Its flexibility, compatibility, and widespread industry use make it a versatile tool for managing diverse datasets.

## Coding

### HTML file code

<!DOCTYPE html>

<html>

    <head>

        <meta name="viewport" content="width=device-width, intital-scale=1.0">

        <style>

            table, th, td

            {

                border-style: solid;

                border-collapse: collapse;

            }

            tr

            {

                font-family: Helvetica

            }

            caption

            {

                font-family: Helvetica

            }

            .p1

            {

                color: aliceblue;

            }

            .p1 {

                color: rgb(5, 21, 35);

            }

           .container {

                column-count: 2;

                column-gap: 100px;

                column-rule-style: solid;

                column-rule-width: 2px;

                column-rule-color: #000000;

            }

           .img2 {

                float: right;

                width: 500px;

                height: 450px;

                padding-bottom: 30%;

                padding-top: 30%;

                background-size: contain;

                background-repeat: no-repeat;

            }

            .letters1 {

                float: left;

            }

            .letters2 {

                float: right;

            }

            p {

                text-indent: 40px;

            }

            p.a {

                text-indent: 470px;

            }

        </style>

    </head>

    <body topmargin="50" leftmargin="50">

        <br>

        <h1 align="center">

            <font face="Cascadia Code SemiBold" size="5">

                <h1><b>News</b></h1>

            </font>

        <hr size="10" noshade color="468284">

    </body>

        <table class="p.solid" id="parts" width="100%" and height="50%">

            <thead>

                <tr>

                    <th>

                        <font face="Cooper Black" size="5">Year</font> </a>

                    </th>

                    <th>

                        <font face="Cooper Black" size="5">Death Rate</font> </a>

                    </th>

                    <th>

                        <font face="Cooper Black" size="5">Growth Rate</font> </a>

                    </th>

                </tr>

            </thead>

        </table>

        <br>

        <hr size="10" noshade color="468284">

        <div id="container"></div>

            <script>

                // 1) creating a new XMLHttpRequest object

                const xhr = new XMLHttpRequest();

                // 2) setting it up to perform a GET request to fetch xml\_content1.xml

                xhr.open('GET', 'xml\_content1.xml', true);

                // 3) When the request is complete, the onload function is called.

                // If the status is between 200 and 399, it means the request was successful.

                // The XML data is then parsed into a DOM object.

                // Retrieving a list of elements with the tag name 'item'.

                // Creating a new table element.

                // Looping through each 'item' element, extract the 'name' and 'description',

                // and create rows and cells for each in the table.

                // The table is appended to the element with the id 'container'.

                xhr.onload = function () {

                    if (xhr.status >= 200 && xhr.status < 400) {

                        const xmlString = xhr.response;

                        const parser = new DOMParser();

                        const xmlDoc = parser.parseFromString(xmlString, 'application/xml');

                        const items = xmlDoc.getElementsByTagName('item');

                        const table = document.createElement('table');

                    for (let i = 0; i < items.length; i++) {

                        const item = items[i];

                        const itemName = item.getElementsByTagName('name')[0].textContent;

                        const itemDescription = item.getElementsByTagName('description')[0].textContent;

                        const row = document.createElement('tr');

                        const nameCell = document.createElement('td');

                        nameCell.textContent = itemName;

                        const descriptionCell = document.createElement('td');

                        descriptionCell.textContent = itemDescription;

                        row.appendChild(nameCell);

                        row.appendChild(descriptionCell);

                        table.appendChild(row);

                    }

                    document.getElementById('container').appendChild(table);

                } else {

                    console.error('Error fetching XML:', xhr.statusText);

                }

            };

            // 4) If there's an error during the request,

            // the onerror function is called, which logs an error message

            xhr.onerror = function () {

                console.error('Error fetching XML:', xhr.statusText);

            };

            xhr.send();

        </script>

</html>

### XML file code

<?xml version="1.0" encoding="UTF-8"?>

<mortality>

    <!-- 1 -->

    <part>

        <year>2023</year>

        <deathrate>7.711</deathrate>

        <growthrate>0.430%</growthrate>

    </part>

    <part>

        <year>2022</year>

        <deathrate>7.678</deathrate>

        <growthrate>0.430%</growthrate>

    </part>

    <part>

        <year>2021</year>

        <deathrate>7.645</deathrate>

        <growthrate>0.430%</growthrate>

    </part>

    <part>

        <year>2020</year>

        <deathrate>7.612</deathrate>

        <growthrate>0.440%</growthrate>

    </part>

    <part>

        <year>2019</year>

        <deathrate>7.579</deathrate>

        <growthrate>0.440%</growthrate>

    </part>

    <part>

        <year>2018</year>

        <deathrate>7.546</deathrate>

        <growthrate>-0.320%</growthrate>

    </part>

    <part>

        <year>2017</year>

        <deathrate>7.570</deathrate>

        <growthrate>-0.320%</growthrate>

    </part>

    <part>

        <year>2016</year>

        <deathrate>7.594</deathrate>

        <growthrate>-0.330%</growthrate>

    </part>

    <part>

        <year>2015</year>

        <deathrate>7.619</deathrate>

        <growthrate>-0.310%</growthrate>

    </part>

    <part>

        <year>2014</year>

        <deathrate>7.643</deathrate>

        <growthrate>-0.310%</growthrate>

    </part>

    <!-- 10 -->

</mortality>