



XML, CSS and Markdown

Alan Shteyman
October 25, 2021

Topics

- What is a Markup Language
- XML, an example Markup Language, allows the internet to look the way it does
 - Some features of XML
 - Formatting the data in an XML file with a CSS file
- Alternative Formatting Frameworks
- Markdown Examples
- Identifying and analyzing file types



Markup Languages

- Markup Languages are computer languages that use tags to define elements within a document.
- They are designed to be both human (H-read) and computer (C-read) readable, though are not a full programming language
- They attempt to make structuring data as easy as possible
 - As well making the data as portable as possible



XML

The image features the letters 'XML' in a large, bold, light blue font with a slight 3D effect. The background is a dark blue grid with various characters including binary digits (0, 1), hexadecimal characters (A, B, C, D, E, F), and other alphanumeric symbols (0, 1, 2, 3, 4, 6, 8, 9) scattered across it, creating a digital or data-themed aesthetic.

Markup Languages (Cont.)

- XML is a type of Markup Language
 - It stands for eXtensible Markup Language
- XML files normally do not contain style or layout instructions
- There are many different standards for XML
 - Documentation can be found online, as well as help for problem solving
- There are both explicit specifications and context clues for the standard for a file



How do Web Pages look the way they do

- Web Pages are more than just text on a background (See *90's example*)
- They are structured to include elements like logos, text, images and video
- XML files commonly contain the data to populate webpages (See *Nextrain example*)

Refs: <http://info.cern.ch/hypertext/WWW/TheProject.html> and <https://nextstrain.org/>

The image compares two websites side-by-side. On the left is the '90's Website' titled 'World Wide Web'. It features a plain white background with blue hyperlinks and a simple text-based layout. On the right is the 'Nextstrain Website', which is modern and visually rich. It has a teal header, a large teal circle with 'VS' in white, and a colorful 'Nextstrain' logo. The content includes a navigation menu (HELP, DOCS, BLOG, LOGIN), a main heading 'Real-time tracking of pathogen evolution', and several content blocks with images and text, such as 'SARS-CoV-2 (COVID-19)' and 'Nextstrain Groups'.

90's Website

World Wide Web

The WorldWideWeb (W3) is a wide-area [hypermedia](#) information retrieval initiative aiming to give universal access to a large universe of documents.

Everything there is online about W3 is linked directly or indirectly to this document, including an [executive summary](#) of the project, [Mailing lists](#) , [Policy](#) , November's [W3 news](#) , [Frequently Asked Questions](#) .

[What's out there?](#)
Pointers to the world's online information, [subjects](#) , [W3 servers](#) , etc.

[Help](#)
on the browser you are using

[Software Products](#)
A list of W3 project components and their current state. (e.g. [Line Mode](#) ,[X11 Viola](#) , [NeXTStep](#) , [Servers](#) , [Tools](#) , [Mail robot](#) , [Library](#))

[Technical](#)
Details of protocols, formats, program internals etc

[Bibliography](#)
Paper documentation on W3 and references.

Nextstrain Website

HELP DOCS BLOG LOGIN

Nextstrain

Real-time tracking of pathogen evolution

Nextstrain is an open-source project to harness the scientific and public health potential of pathogen genome data. We provide a continually-updated view of publicly available data alongside powerful analytic and visualization tools for use by the community. Our goal is to aid epidemiological understanding and improve outbreak response. If you have any questions, or simply want to say hi, please give us a shout at hello@nextstrain.org.

[READ MORE](#)

SARS-CoV-2 (COVID-19)

We are incorporating SARS-CoV-2 genomes as soon as they are shared and providing analyses and situation reports. In addition we have developed a number of resources and tools, and are facilitating independent groups to run their own analyses.

Nextstrain Groups

We want to enable research labs, public health entities and others to share their datasets and narratives through Nextstrain with complete control of their data and audience.

Latest global analysis - GISAID data | Latest global analysis - open data

neherlab | spheres

Structure of a Sample Basic XML file

We will discuss the following elements:

- Header
- Comments
- Root Element
- Child Element
- Nested Child Elements
- Extensibility
(designed to allow the addition of new capabilities and functionality)
- Tree Structure
- Interaction with CSS



Header

```
<?xml version="1.0" encoding='utf-8'?>
```

Comments

```
<?xml version="1.0" encoding='utf-8'?>
```

```
<!-- A SAMPLE XML data Structure – This is a comment -->
```

```
<!-- Above is a header, below is the data -->
```


Root Element

```
<?xml version="1.0" encoding='utf-8'?>
```

```
<!-- A SAMPLE XML data Structure – This is a comment -->
```

```
<!-- Above is a header, below is the data -->
```

```
<Company>
```

```
</Company>
```

Child Element

```
<?xml version="1.0" encoding='utf-8'?>
```

```
<!-- A SAMPLE XML data Structure – This is a comment -->
```

```
<!-- Above is a header, below is the data -->
```

```
<Company>
```

```
<Employee>
```

```
</Employee>
```

```
</Company>
```

Multiple Nested Child Elements

```
<?xml version="1.0" encoding='utf-8'?>
```

```
<!-- A SAMPLE XML data Structure – This is a comment -->
```

```
<!-- Above is a header, below is the data -->
```

```
<Company>
```

```
  <Employee>
```

```
    <FirstName>Tanmay</FirstName>
```

```
    <LastName>Patil</LastName>
```

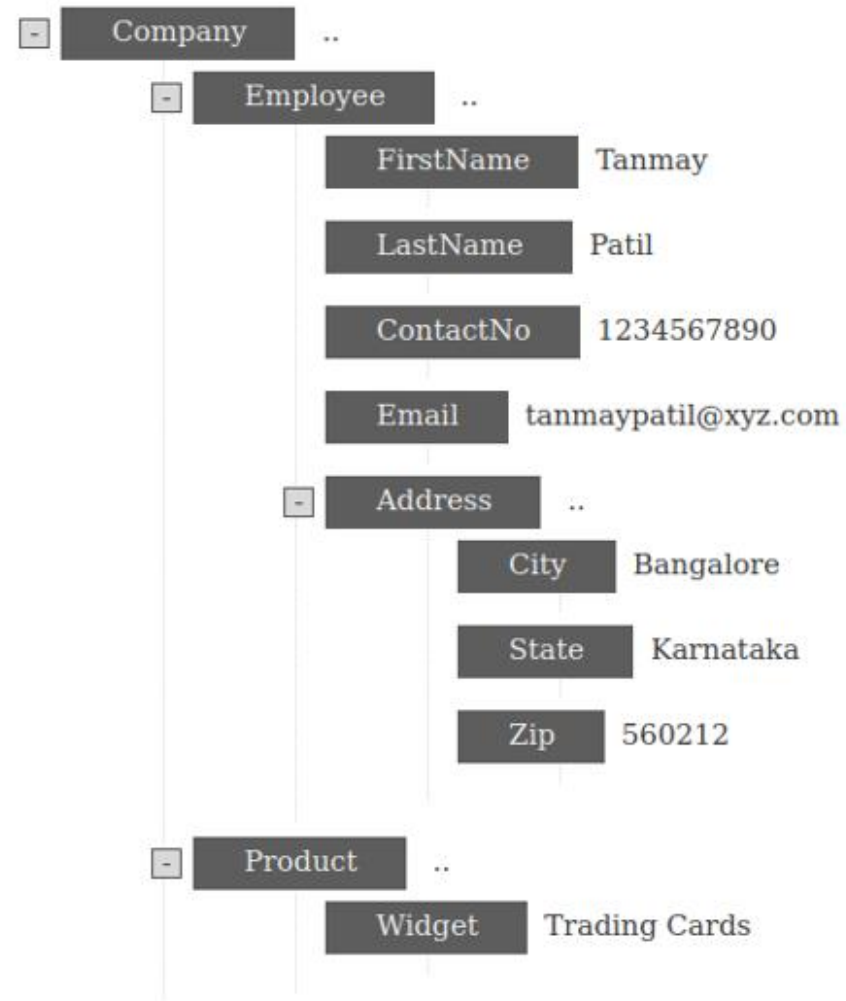
```
  </Employee>
```

```
</Company>
```

The number of Elements is Extensible

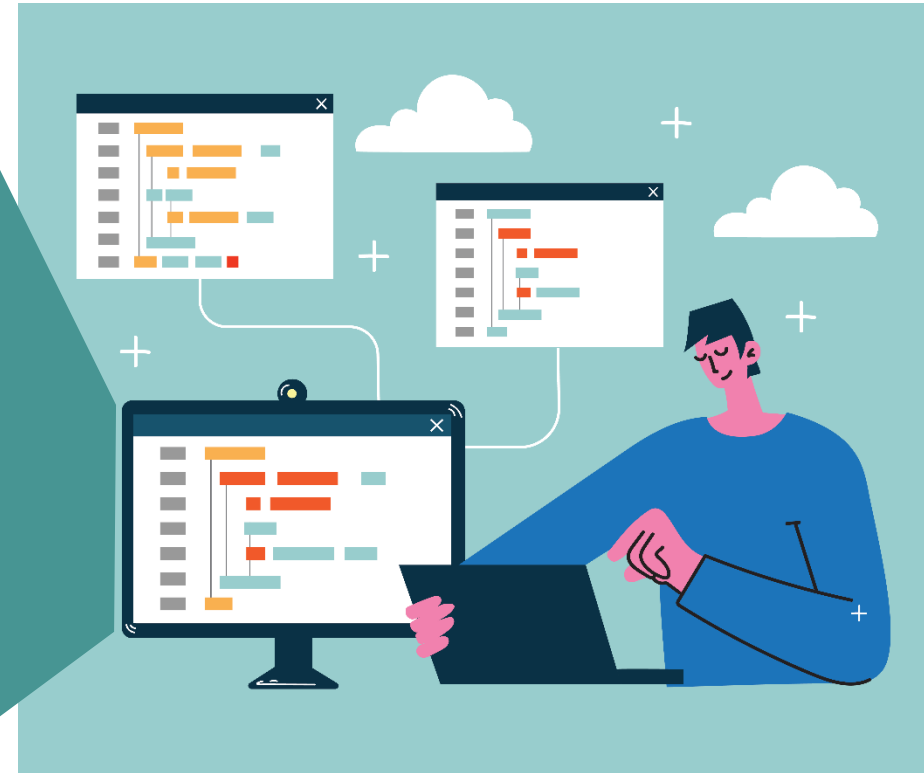
```
<?xml version="1.0" encoding='utf-8'?>  
<!-- A SAMPLE XML data Structure – This is a comment -->  
<!-- Above is a header, below is the data -->  
<Company>  
  <Employee>  
    <FirstName>Tanmay</FirstName>  
    <LastName>Patil</LastName>  
    <ContactNo>1234567890</ContactNo>  
    <Email>tanmaypatil@xyz.com</Email>  
  </Employee>  
</Company>
```

Representation of the Tree Structure of the Data



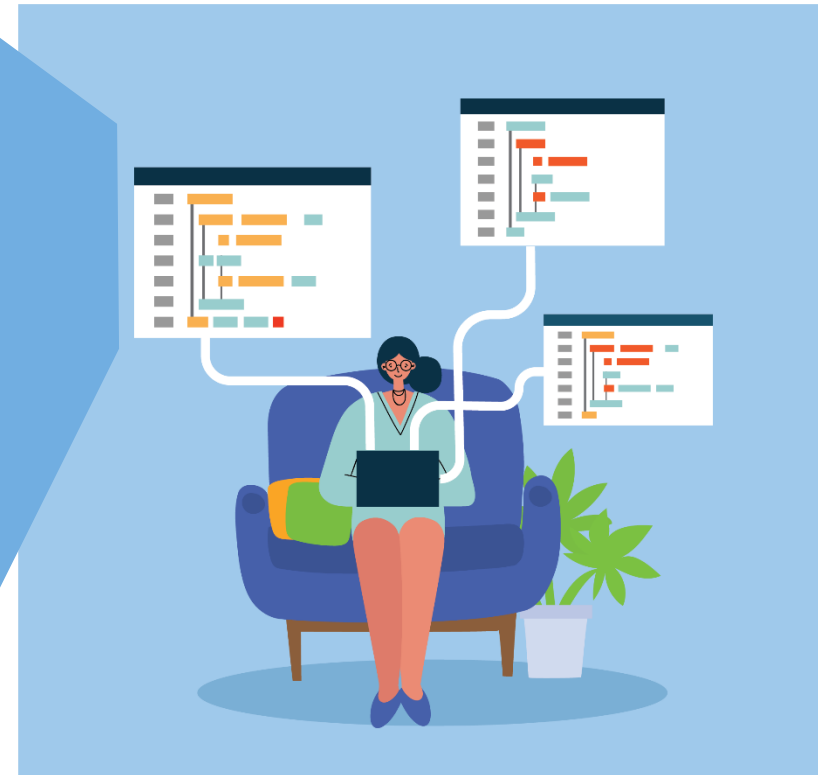
Can have a sophisticated Tree Structure

```
<?xml version="1.0" encoding='utf-8'?>
<!-- A SAMPLE XML data Structure – This is a comment -->
<!-- Above is a header, below is the data -->
<Company>
  <Employee>
    <FirstName>Tanmay</FirstName>
    <LastName>Patil</LastName>
    <ContactNo>1234567890</ContactNo>
    <Email>tanmaypatil@xyz.com</Email>
    <Address>
      <City>Bangalore</City>
      <State>Karnataka</State>
      <Zip>560212</Zip>
    </Address>
  </Employee>
</Company>
```



Can have any Tree structure

```
<?xml version="1.0" encoding='utf-8'?>
<!-- A SAMPLE XML data Structure – This is a comment -->
<!-- Above is a header, below is the data -->
<Company>
  <Employee>
    <FirstName>Tanmay</FirstName>
    <LastName>Patil</LastName>
    <ContactNo>1234567890</ContactNo>
    <Email>tanmaypatil@xyz.com</Email>
    <Address>
      <City>Bangalore</City>
      <State>Karnataka</State>
      <Zip>560212</Zip>
    </Address>
  </Employee>
  <Product>
    <Widget>Trading Cards</Widget>
  </Product>
</Company>
```



Add a Header to interact with CSS

```
<?xml version="1.0" encoding='utf-8'?>
```

```
<?xml-stylesheet type="text/css" href="Rule.css"?>
```

```
<!-- below the version header is the xml-stylesheet which allows the use of a css file to format how the xml file is presented -->
```

```
<!-- A SAMPLE XML data Structure – This is a comment -->
```

```
<!-- Above is a header, below is the data -->
```

```
<Company>
```

```
  <heading>Welcome to the Company </heading>
```

```
  <heading>Here are the current Employees </heading>
```

```
<Employee>
```

```
  <FirstName>Tanmay</FirstName>
```

```
  <LastName>Patil</LastName>
```

```
  <ContactNo>1234567890</ContactNo>
```

```
  <Email>tanmaypatil@xyz.com</Email>
```

Notice

The extra header line specifying that this .xml file is meant to be used along with a .css file. Extra information is added in the form of headings which are basically just another example of an element

Add a Header to interact with CSS (Cont.)

```
<Email>tanmaypatil@xyz.com</Email>  
  <Address>  
    <City>Bangalore</City>  
    <State>Karnataka</State>  
    <Zip>560212</Zip>  
  </Address>  
</Employee>  
<heading>Here are the Products we make </heading>  
  <Product>  
    <Widget>Trading Cards</Widget>  
  </Product>  
  <pic1>  
  </pic1>  
</Company>
```

Notice

The `<pic1>` tags, these tags state an element `pic1` will go here. In the example, we will eventually put an image there, although it is laid out here as an empty element.

What is CSS

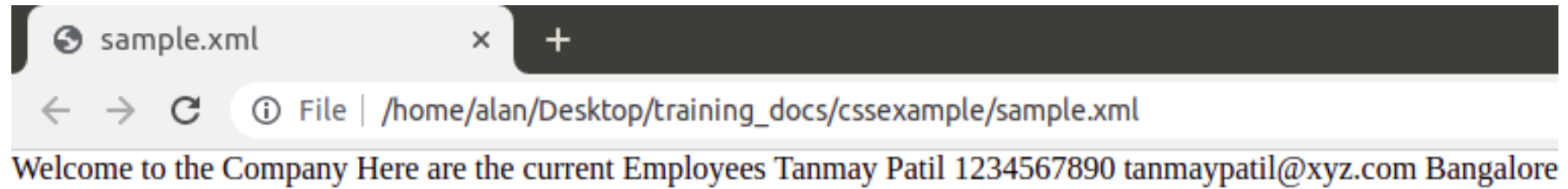
- CSS (Cascading Style Sheets) is a language for specifying how documents are presented to users — how they are styled, laid out, etc.
- Together CSS and XML not only present data to a user but also stylize the presentation
- The end user usually sees the results in a web browser



Sample Basic CSS file

```
heading {color: black;}
```

Data with no Style



NOTICE

All the information is here
but all on one line

Basic CSS heading formatting file

```
heading {color: green;  
        font-size : 80px;  
        background-color : blue;}
```

Data with only Heading Formatted



Adding style changes to the CSS file change how the Headings are shown specifically

Basic CSS Child element style file

```
Employee {color: red;  
          font-size : 20px;  
          background-color : yellow;}
```

Data with Style for one Element



Now only the Employee information is styled

CSS two Child element style file

```
Employee {color: red;  
          font-size : 20px;  
          background-color : yellow;}
```

```
Product {color: purple;  
         font-size : 40px;  
         background-color : gray;}
```

Data with Style for Two Element



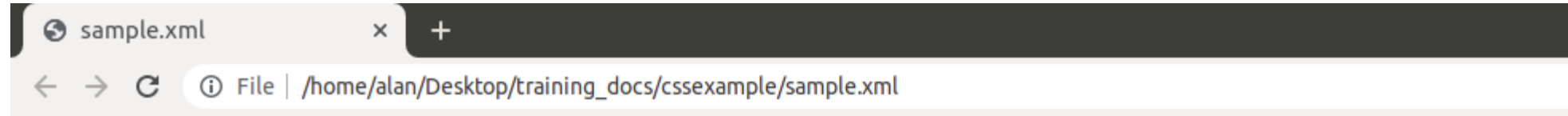
↑ ↑

Here we can see different elements styled differently – different style specifications can stack and be made to be as different as desired

CSS Basic space formatting file

heading, Employee, Product, FirstName, Lastname, ContactNo,
Email, Address { display : block;}

Data with style for positioning



Welcome to the Company
Here are the current Employees
Tanmay
Patil
1234567890
tanmaypatil@xyz.com
Bangalore Karnataka 560212
Here are the Products we make
Trading Cards

Here is a basic example of formatting that actually changes how the information is spatially laid out – Different elements and different headers are now on different lines



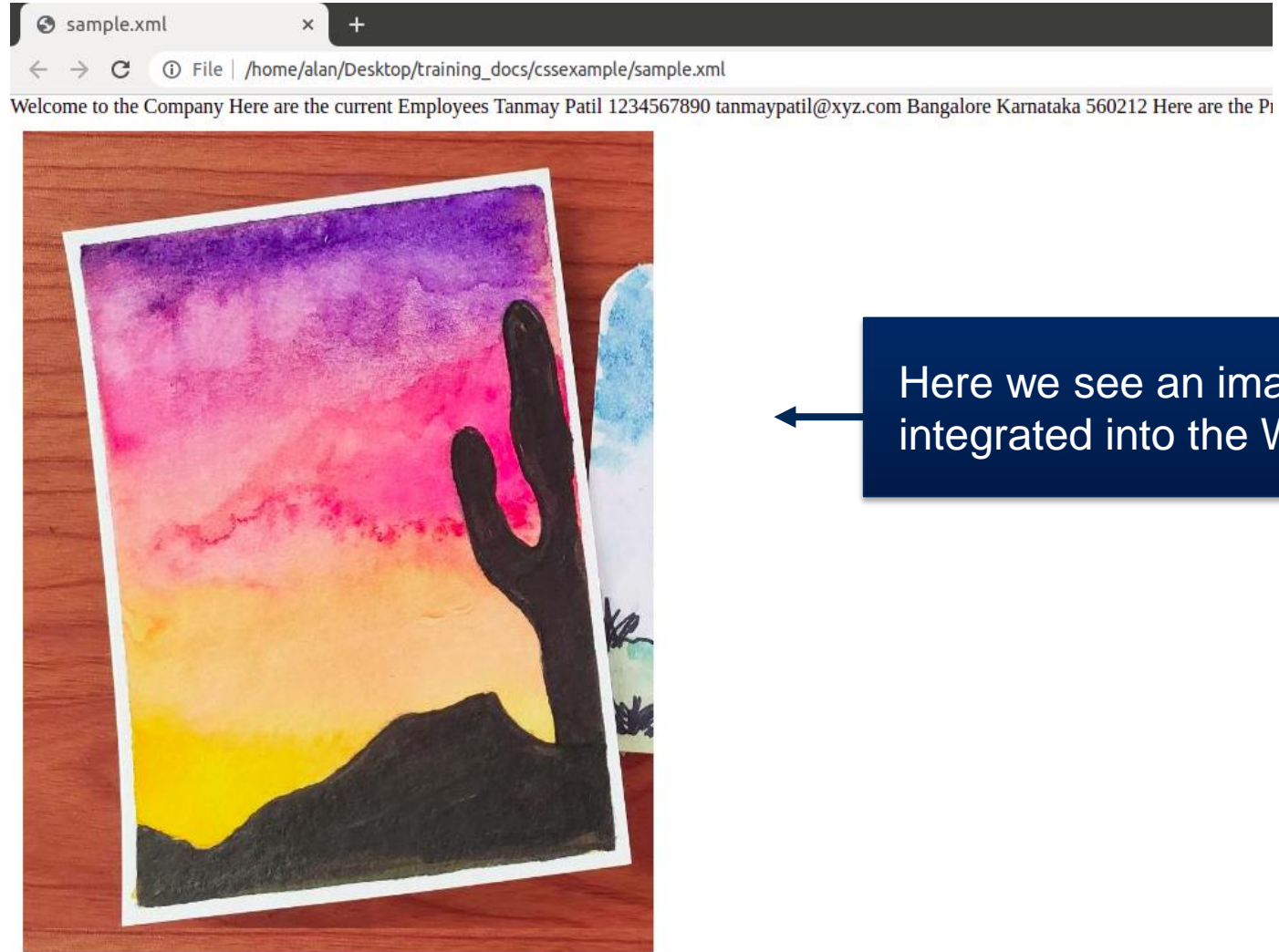
CSS to present an image

```
pic1 {display: block;
background-image: url('Trading-Cards.jpg');
margin: 10px;
width: 491px;
height: 640px;
}
```

Notice

The path to the image [inside the url() function] has to be specified as well as tell the Web Page how much of the image to show. The width and the height are in pixels and to show a full image or a partial image the height and width must be adjusted.

Add an Image to the Data



← Here we see an image being integrated into the Web page

CSS file – putting it all together

```
heading {color: green;  
    font-size : 80px;  
    background-color : blue;}
```

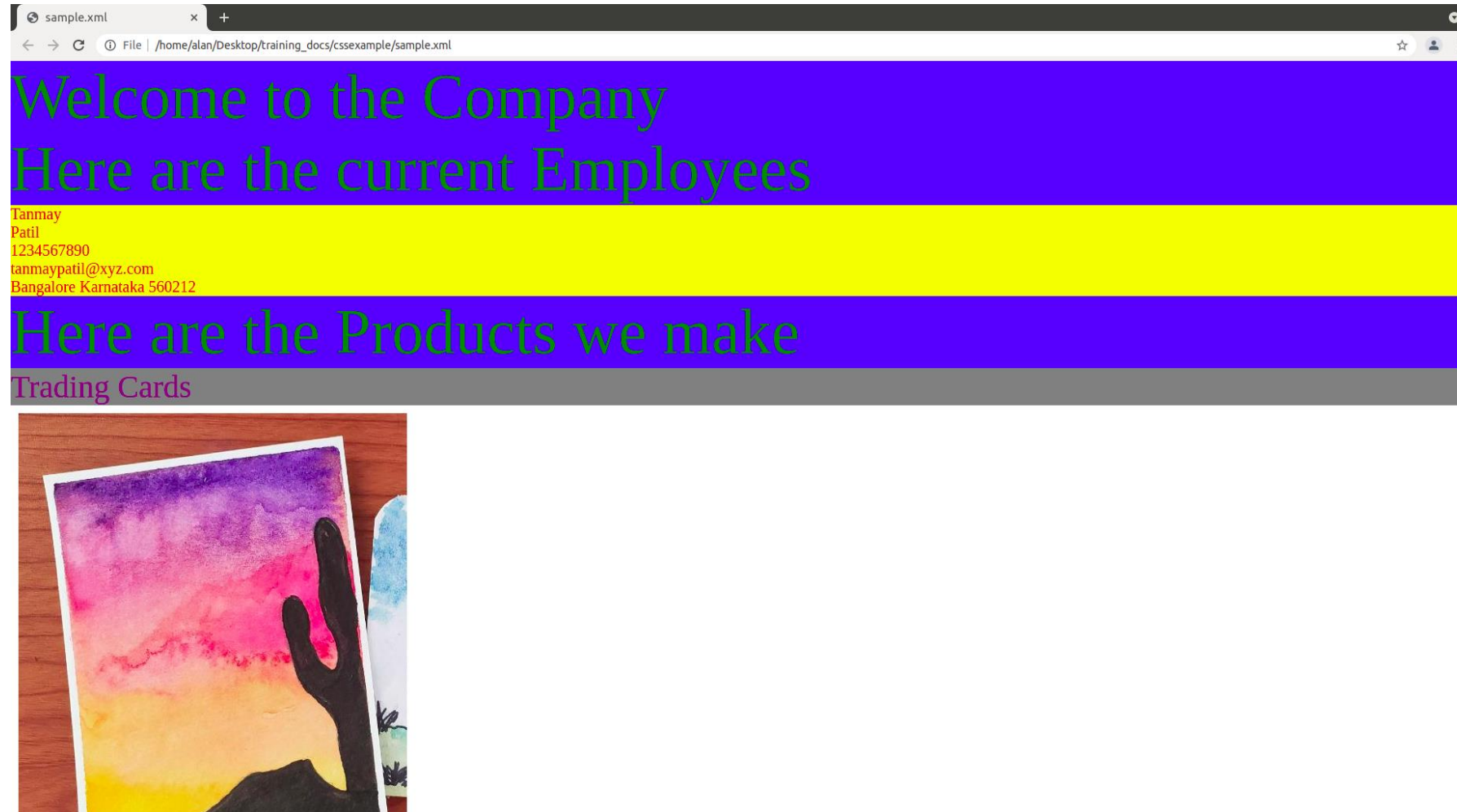
```
Employee {color: red;  
    font-size : 20px;  
    background-color : yellow;}
```

```
Product {color: purple;  
    font-size : 40px;  
    background-color : gray;}
```

```
heading, Employee, Product, FirstName, Lastname, ContactNo, Email, Address { display : block;}
```

```
pic1 {display: block;  
    background-image: url('Trading-Cards.jpg');  
    margin: 10px;  
    width: 491px;  
    height: 640px;}
```

Data with Complex Formatting



Mixing and matching and stacking different types of formatting changes can make a website look potentially like anything

Alternative Frameworks to structure Web Pages

- In addition to CSS, there are other ways to format data in web pages
 - HTML (Hyper Text Markup Language) is a widely used framework
 - Markdown is a lightweight markup language for creating formatted text using a plain-text editor

Alternative Frameworks to structure Web Pages (Cont.)

- HTML can do similar things to CSS, but using a different behind-the-scenes framework
 - In the interest of time, no further description of HTML will be done here
- Markdown is lighter than HTML or CSS, but many more tools and languages can utilize it as is. For example:
 - Python, Java, R, and others have built in utilities for Markdown
- It has simpler formatting compared to the other languages mentioned here
 - I.E. fewer keystrokes per formatting change, fewer files to keep track of (.md),etc.

Basic text file with Markdown decorators

An h1 header

```
=====
```

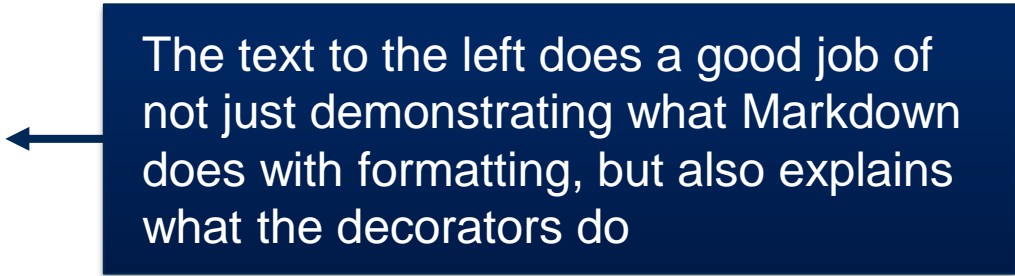
Paragraphs are separated by a blank line.

2nd paragraph. **Italic**, ****bold****, and ``monospace``. Itemized lists look like:

- * this one
- * that one
- * the other one

Note that --- not considering the asterisk --- the actual text content starts at 4-columns in.

(Example taken from <http://www.unexpected-vortices.com/sw/rippledoc/quick-markdown-example.html>)



The text to the left does a good job of not just demonstrating what Markdown does with formatting, but also explains what the decorators do

Sample Markdown result

An h1 header

Paragraphs are separated by a blank line.

2nd paragraph. *Italic*, **bold**, and `monospace`. Itemized lists look like:

- this one
- that one
- the other one

Note that — not considering the asterisk — the actual text content starts at 4-columns in.

Basic text file with Markdown decorators

- > Block quotes are
- > written like so.
- >
- > They can span multiple paragraphs,
- > if you like.

Use 3 dashes for an em-dash. Use 2 dashes for ranges (ex., "it's all in chapters 12--14"). Three dots ... will be converted to an ellipsis.

Unicode is supported. 😊

Sample Markdown result

Block quotes are written like so.

They can span multiple paragraphs, if you like.

Use 3 dashes for an em-dash. Use 2 dashes for ranges (ex., “it’s all in chapters 12-14”). Three dots ... will be converted to an ellipsis. Unicode is supported. 😊

Basic text file with Markdown decorators

An h2 header

Here's a numbered list:

1. first item
2. second item
3. third item

Note again how the actual text starts at 4 columns in (4 characters from the left side). Here's a code sample:

```
# Let me re-iterate ...  
for i in 1 .. 10 { do-something(i) }
```

As you probably guessed, indented 4 spaces. By the way, instead of indenting the block, you can use delimited blocks, if you like:

Sample Markdown result

An h2 header

Here's a numbered list:

1. first item
2. second item
3. third item

Note again how the actual text starts at 4 columns in (4 characters from the left side). Here's a code sample:

```
# Let me re-iterate ...  
for i in 1 .. 10 { do-something(i) }
```

As you probably guessed, indented 4 spaces. By the way, instead of indenting the block, you can use delimited blocks, if you like:

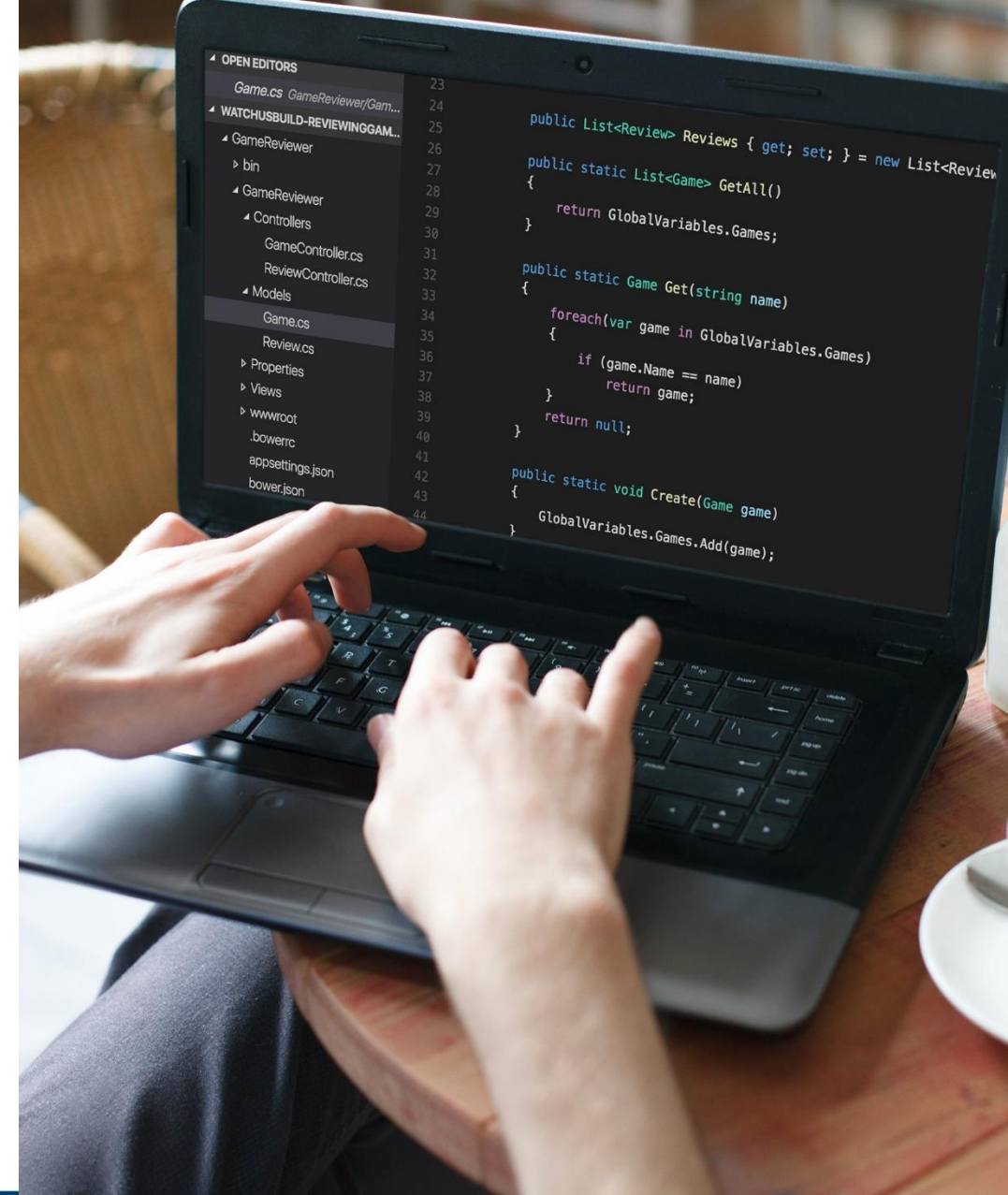
Markdown has a lot of Features

- This presentation has demonstrated a short list of the ways you can use Markdown to format text
- There is a more extensive example at <http://www.unexpected-vortices.com/sw/rippledoc/quick-markdown-example.html>
- There are many Markdown parsers to use
 - Python (markdown), R (Rmarkdown)

A Language Parser is a program that reads in code for a language and interprets it to convert and execute it

Recognizing and understanding file types

- On the current internet, there are many different frameworks that can do the same thing
 - For example CSS/XML vs HTML
- And for each framework, there are multiple different versions being used



Recognizing and understanding file types (cont.)

- Understanding what a file type is and how to use it with program usually is done with a web search
 - Manuals, Tutorials, Examples, Video Walk-Throughs, etc.
- However, a fair bit about files can be understood in general by looking at trends in the files themselves



Recognizing and understanding file types (cont.)

- Binary files
 - Meant to be C-read and not H-read
 - Tend to need less space than H-read files
 - Often can be processed faster by a computer than H-read files
- It is usually not worth it to try to understand in a Binary file directly
 - Converting a Binary file to a H-read file is usually more effective



Recognizing and understanding file types (cont.)

- In general H-read files tend to follow certain patterns
 - They may or may not have headers
 - They may have repeating structures – recursive vs iterative
 - Lines from iterative files tend to be organized in columns
 - The suffix of a file can give hint at to the file type (.xml,.css,.md vs .txt)



Sample .xml file

```
<?xml version="1.0" encoding='utf-8'?>
```

```
<!-- A SAMPLE XML data Structure – This is a comment -->
```

```
<!-- Above is a header, below is the data -->
```

```
<Company>
```

```
  <Employee>
```

```
    <FirstName>Tanmay</FirstName>
```

```
    <LastName>Patil</LastName>
```

```
    <ContactNo>1234567890</ContactNo>
```

```
    <Email>tanmaypatil@xyz.com</Email>
```

```
  </Employee>
```

```
</Company>
```

Notice

how this file type has a header, comments and stores data in a recursive fashion. The header gives hints about what type of file this is.

Sample .css file

```
heading {color: green;
        font-size : 80px;
        background-color : blue;}

Employee {color: red;
        font-size : 20px;
        background-color : yellow;}

Product {color: purple;
        font-size : 40px;
        background-color : gray;}

heading, Employee, Product, FirstName,Lastname, ContactNo, Email, Address { display : block;}

pic1 {display: block;
      background-image: url('Trading-Cards.jpg');
      margin: 10px;
      width: 491px;
      height: 640px;}
```

Notice

This file has no header and while individual style changes appear iteratively, complex specifications can be nested inside each item in a recursive manner. Space between each item purely for ease of reading and there is much leeway in how these entries can be formatted.

Sample .tsv file

Notice

how this file is strictly iterative and has a header which describes what each column is describing. The actual file has lines separate by newline characters and columns separated by tab characters and the data is shown here as a table for clarity.

In this context an iterative pattern is one that repeats while holding different data (a file that has well defined rows like below, is iterative).

NAME	TAXID	READ_COUNT	READ_COUNT_RNR
Clostr. botulinum	1491	10	5
Hala. praevalens	2331	100	20
Bacillus amyloliquefaciens	1390	1000	990



The science you expect.
The people you know.

THANK YOU

Follow us and learn more
about our work

 [Facebook.com/MRIGlobalResearch](https://www.facebook.com/MRIGlobalResearch)

 [Youtube.com/user/MRIExternalComm](https://www.youtube.com/user/MRIExternalComm)

 [Linkedin.com/company/mriglobal](https://www.linkedin.com/company/mriglobal)

 twitter.com/MRIGlobal